Beyond the tracked line of sight: Gaze-driven user models

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard-Karls-Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von
David Geisler
aus Weingarten

Tübingen
2020

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation:	22.12.2020
Stellvertretender Dekan:	Prof. Dr. József Fortágh
1. Berichterstatter:	Prof. Dr. Enkelejda Kasneci
2. Berichterstatter:	Prof. Dr. Oliver Bringman

Abstract

The human visual perception is one of the most significant but limited sensory input. A small nerve tissue – the retina – on the back on an optical apparatus – the eyeball – provides a continuous flow of visual information of our environment. Yet, the retinal visual perception capabilities differ significantly depending on the affected region. In a nutshell, a small region in the center of the retina – the fovea – provides high acuity and the maximal perceivable color spectrum. However, with increasing eccentricity, the retinal visual perception becomes successively less sensitive to spatial and spectral contrasts. This provides humans with a highly developed foveal perception, yet limited to a small area in the field of view. To overcome this limitation, humans move their eyes, rapidly shifting their visual attention towards significant scene areas. The ensuing eye movements can be recorded and evaluated by state-of-the-art video-based eye-tracking systems. This thesis presents a full hardware and software stack to record the user's pupil position and to predict the user's line of sight – the imaginary line between focused scene point and fovea – in 3D.

However, the tracked line of sight merely provides the alignment of the user's foveal attention. Yet, human visual perception goes far behind a straight line. Although the fovea offers the highest visual capacity and generally coincides with the center of visual attention, foveal perception is neither sufficiently spatially characterized by a line of sight nor is visual perception limited to the fovea. Rather, the entire visual field is continuously and subliminally scanned and reconciled with a mental image of the scene. Thus, highly developed and sophisticated cognitive processes equalize the received visual stimulus with the anticipated scene image. Hence, retinal extracted visual features are differently emphasized, suppressed, recombined, and abstracted, depending on the performed task, intention, experience, fatigue, and many other factors. This reciprocal process of retinal visual feature extraction (bottom-up) and the selection and interpretation of higher cognitive processes (top-down) is the principle of fixation target identification and consciously perceived scene content. Thus, to assess the user's visual perception, it is crucial to evaluate the user's whole field of view and to incorporate the visual stimulus, retinal capability, and cognitive state.

This thesis provides novel gaze-driven user models to predict the retinal extracted visual stimuli over the entire field of view and the user's cognitive situation. This comprehensive evaluation of the user's visual perception beyond the mere tracked line of sight enables sophisticated Human-Computer Interaction (HCI) and Human-Robot Interaction (HRI) applications that integrate the user's visual perception and situation awareness into their interfaces, to enable natural, user-centric, and intelligent collaboration systems.

Zusammenfassung

Die visuelle Wahrnehmung des Menschen ist einer der bedeutendsten, aber dennoch begrenzten sensorischen Inputs. Ein kleines Nervengewebe - die Netzhaut - auf der Rückseite eines optischen Apparates - des Augapfels - sorgt für einen kontinuierlichen Fluss visueller Informationen über unsere Umwelt. Die visuellen Wahrnehmungsfähigkeiten der Netzhaut unterscheiden sich jedoch je nach betroffener Region erheblich. Kurz gesagt, eine kleine Region in der Mitte der Netzhaut - die Fovea - sorgt für hohe Sehschärfe und das maximal wahrnehmbare Farbspektrum. Mit zunehmender Exzentrizität wird die visuelle Wahrnehmung der Netzhaut jedoch sukzessive unempfindlicher gegenüber räumlichen und spektralen Kontrasten. Dies ermöglicht dem Menschen eine hochentwickelte foveale Wahrnehmung, die jedoch auf einen kleinen Bereich im Gesichtsfeld beschränkt ist. Um diese Beschränkung zu überwinden, bewegt der Mensch jedoch seine Augen und kann so seine visuelle Aufmerksamkeit schnell und sequentiell auf bedeutende Szenenbereiche verlagern. Die sich daraus ergebenden Augenbewegungen können mit modernsten videobasierten Blickverfolgungssystemen aufgezeichnet und ausgewertet werden. In dieser Arbeit wird ein vollständiger Hard- und Softwarestapel vorgestellt, um die Pupillenposition des Benutzers aufzuzeichnen und die Sichtlinie des Benutzers - die imaginäre Linie zwischen fokussiertem Szenenpunkt und Fovea - in 3D vorherzusagen.

Die getrackte Sichtlinie liefert jedoch lediglich die Ausrichtung der fovealen Aufmerksamkeit des Benutzers. Die visuelle Wahrnehmung des Menschen bleibt jedoch weit hinter einer geraden Linie zurück. Obwohl die Fovea die höchste visuelle Kapazität bietet und im Allgemeinen mit dem Zentrum der visuellen Aufmerksamkeit zusammenfällt, ist die foveale Wahrnehmung weder räumlich ausreichend durch eine Sichtlinie charakterisiert, noch ist die visuelle Wahrnehmung auf die Fovea beschränkt. Vielmehr wird das gesamte Gesichtsfeld kontinuierlich unterschwellig abgetastet und mit einem imaginären Bild der Szene in Einklang gebracht. So gleichen hochentwickelte und anspruchsvolle kognitive Prozesse den empfangenen visuellen Reiz mit dem erwarteten Szenenbild aus. Abhängig von der ausgeführten Aufgabe, der Absicht, der Erfahrung, der Müdigkeit und vielen anderen Faktoren werden verschiedene retinal extrahierte visuelle Merkmale betont, unterdrückt, neu kombiniert und abstrahiert. Dieser reziproke Prozess der Extraktion retinaler visueller Merkmale (bottom-up) und der Auswahl und Interpretation höherer kognitiver Prozesse (top-down) ist das Prinzip der Fixationszielidentifikation und des bewusst wahrgenommenen Szeneninhalts. Um die visuelle Wahrnehmung des Benutzers zu beurteilen, ist es daher entscheidend, das gesamte Gesichtsfeld des Benutzers zu bewerten und den visuellen Stimulus, die retinale Fähigkeit und die kognitive Situation einzubeziehen.

Diese Arbeit stellt neuartige blickgesteuerte Benutzermodelle zur Verfügung, um die von der Netzhaut extrahierten visuellen Stimuli über das gesamte Gesichtsfeld und die kognitive Situation des Benutzers vorherzusagen. Diese umfassende Evaluierung der visuellen Wahrnehmung des Benutzers über die bloße verfolgte Blickrichtung hinaus ermöglicht anspruchsvolle HCI- und HRI-Anwendungen, die die visuelle Wahrnehmung des Benutzers und sein Situationsbewusstsein in ihre Schnittstellen integrieren, um natürliche, benutzerzentrierte und intelligente Kollaborationssysteme zu ermöglichen.

Contents

Preface **1**

1 Introduction **1**
1.1 Motivation 2
1.2 Scope of the thesis 3
1.3 Structure of this thesis & scientific contribution 3

2 Fundamentals **8**
2.1 Eye-tracking 8
2.2 Head-mounted eye-tracking 11
2.2.1 Hardware 12
2.2.2 Pupil detection 13
2.2.3 Gaze point 16
2.2.4 Processing the line of sight 26
2.2.5 Remarks 29
2.3 Remote eye-tracking 30
2.3.1 Hardware 32
2.3.2 Glints 33
2.3.3 Pupil detection 36
2.3.4 Process the line of sight 37
2.3.5 Remarks 37
2.4 Retinal processing 38
2.4.1 Visual saliency 38
2.4.2 Retinal blueprint 41
2.4.3 Retina model 48
2.5 Cognitive model 50
2.5.1 Cognitive eye features 51
2.5.2 Scanpath 54

I Line of sight **58**

3 Head-mounted eye-tracking **60**
3.1 Circular binary features for robust and real-time pupil center detection 61
3.1.1 Method 62
3.1.2 Evaluation 65
3.1.3 Limitations 66
3.1.4 Conclusion 67

3.2 Cycle GANs for pupil and eyelid segmentation 69
3.2.1 Method 70
3.2.2 Evaluation 74
3.2.3 Limitations 78
3.2.4 Conclusion 79

4 Remote eye-tracking **80**
4.1 Realtime glint detection in remote eye-tracking 81
4.1.1 Method 81
4.1.2 Evaluation 89
4.1.3 Limitations 92
4.1.4 Conclusion 92

4.2 Evaluation of state-of-the-art pupil detection for remote eye-tracking 93
4.2.1 Method 94
4.2.2 Evaluation 96
4.2.3 Limitations 99
4.2.4 Conclusion 99

II Retinal processing **100**

5 Visual saliency **101**
5.1 Saliency sandbox . . . 102
5.1.1 Architecture . . . 103
5.1.2 Algorithms . . . 104
5.1.3 Usage . . . 113
5.1.4 Limitations . . . 115
5.1.5 Conclusion . . . 115

5.2 Gaze-driven GBVS saliency map . . . 116
5.2.1 Method . . . 117
5.2.2 Evaluation . . . 120
5.2.3 Limitations . . . 122
5.2.4 Conclusion . . . 123

6 Retinal model **124**
6.1 Biophysiological model . . . 125
6.1.1 Method . . . 126
6.1.2 Evaluation . . . 142
6.1.3 Limitations . . . 146
6.1.4 Conclusion . . . 146

6.2 Functional model . . . 147
6.2.1 Method . . . 148
6.2.2 Evaluation . . . 154
6.2.3 Limitations . . . 163
6.2.4 Conclusion . . . 164

III Cognitive model **165**

7 Cognitive eye features **167**
7.1 Realtime assessment of cognitive load 168
7.1.1 Method 169
7.1.2 Evaluation 172
7.1.3 Limitations 174
7.1.4 Conclusion 174

7.2 Adaptive eye movement classification 175
7.2.1 Method 176
7.2.2 Evaluation 179
7.2.3 Limitations 181
7.2.4 Conclusion 182

8 Scanpath **183**
8.1 Online recognition of driver-activity 184
8.1.1 Method 185
8.1.2 Evaluation 189
8.1.3 Limitations 191
8.1.4 Conclusion 192

8.2 A MinHash approach for fast scanpath classification 193
8.2.1 Method 193
8.2.2 Evaluation 200
8.2.3 Limitations 205
8.2.4 Conclusion 205

Final remarks **206**
8.3 Perception sandbox 206

9 Conclusion and outlook **208**

Postface **213**

Bibliography **213**
1st author publications 214
2nd author publications 215
Other author publications 216
References 217
Online resources 237

Appendix **238**

Figures

Introduction
Structure of this thesis
1.3.1 Thesis roadmap 4

Fundamentals
Eye-tracking
2.1.1 EOG setup 9
2.1.2 Scleral coil tracking 10
2.2.1 Video-based head-mounted eye-tracking 11
Head-mounted eye-tracking
2.2.2 Datasets 15
2.2.3 Pupil detection rates 16
2.2.4 2D calibration function 17
2.2.5 Corneal refraction 19
2.2.6 Polynomial mapping error 20
2.2.7 Polynomial extrapolation error 22
2.2.8 CalibMe compared to 9-point calibration 24
2.2.9 GRIP's ambiguous solution problem 25
2.2.10 ArUCO vehicle setup 27
2.2.11 Line of sight and object's eccentricity processing 28
2.2.12 ArUCO based line of sight processing 29
Remote eye-tracking
2.3.1 Remote eye-tracking setup 31
2.3.2 Glint based geometrical eye model 33
Retinal processing
2.4.1 Top-down saliency map 40
2.4.2 Structure of the retina 42
2.4.3 Rod photoreceptor topography 43
2.4.4 Photoreceptor sensitivity 44
2.4.5 Cone photoreceptor topography 45
2.4.6 RGC topography 49

2.4.7 RGC dendritic field size 50
Cognitive model
2.5.1 Gaze velocity signal and classification 52
2.5.2 Pupil diameter while driving 55
2.5.3 The visual field of view 59

Head-mounted eye-tracking
Circular binary features for robust and real-time pupil center detection
3.1.1 Example of a pupil center probability lookup 62
3.1.2 Feature construction 63
3.1.3 Pupil center detection rate 66
Cycle GANs for pupil and eyelid segmentation
3.2.1 Cycle GAN architecture 72
3.2.2 Segmented eye images 73
3.2.3 Generated eye images 73
3.2.4 Image augmentation 74
3.2.5 Dataset 75

Remote eye-tracking
Realtime glint detection in remote eye-tracking
4.1.1 Experiment setup 85
4.1.2 Dataset overview 90
4.1.3 Dataset examples 91
Evaluation of state-of-the-art pupil detection for remote eye-tracking
4.2.1 New dataset pupil detection challenges 95
4.2.2 Dataset 95
4.2.3 HC detection rate 96
4.2.4 Pupil detection rate 98

Visual saliency
Saliency sandbox
5.1.1 Itti & Koch implementation example 104
5.1.2 Framework architecture 105
5.1.3 Provided modules 106
5.1.4 Input stimulus 107
Gaze-driven GBVS saliency map
5.2.1 Parameters influence 119
5.2.2 ETRA 2020's *Call for Papers* example 121
5.2.3 Yarbus example 121
5.2.4 *Big Buck Bunny* example 122

Retinal model
Biophysiological model
6.1.1 Photoreceptor cell signal propagation . . . 128
6.1.2 HC signal propagation . . . 129
6.1.3 BC signal propagation . . . 130
6.1.4 RGC signal propagation . . . 131
6.1.5 Photoreceptor temporal signal . . . 133
6.1.6 Photoreceptor spatial-temporal signal . . . 134
6.1.7 HC temporal signal . . . 135
6.1.8 HC spatial-temporal signal . . . 135
6.1.9 BC temporal signal . . . 137
6.1.10 BC spatial-temporal signal . . . 137
6.1.11 RGC temporal signal . . . 138
6.1.12 RGC spatial-temporal signal . . . 140
6.1.13 Combined RGC spatial-temporal signal . . . 141
6.1.14 Framework architecture . . . 141
6.1.15 Retinal response example . . . 144
6.1.16 Maximal retinal response distribution . . . 144
6.1.17 Confusion matrix . . . 145
6.1.18 Time related classification accuracy . . . 145
Functional model
6.2.1 Spatio-temporal receptive field convolution kernel . . . 152
6.2.2 Color channel buffers . . . 153
6.2.3 Temporal color contrast . . . 153
6.2.4 Workflow of the filtering pipeline . . . 155
6.2.5 Experiment design . . . 157
6.2.6 NN architecture . . . 161
6.2.7 Classification performance . . . 163

Cognitive eye features
Realtime assessment of cognitive load
7.1.1 Hardware setup and virtual driving scene . . . 169
7.1.2 Driving scene examples . . . 170
7.1.3 Pedestrian warning cues . . . 170
7.1.4 Closest distance and pupil diameter . . . 171
Adaptive eye movement classification
7.2.1 MERCY's architecture . . . 177
7.2.2 GMM eye movement distribution . . . 177
7.2.3 Benchmark of MERCY and BMM . . . 179
7.2.4 Distribution of the average intersection point δ . . . 182

Scanpath
Online recognition of driver-activity

8.1.1 Workflow of SubsMatch based task classification . . . 185
8.1.2 Scanpath distance matrix . . . 188
8.1.3 Handheld vs. hands-free and task classification performance . . . 190
8.1.4 Task classification performance . . . 190
8.1.5 Feature selection . . . 191
A MinHash approach for fast scanpath classification
8.2.1 Workflow of MinHash based scanpath classification . . . 194
8.2.2 Scanpath quantization . . . 195
8.2.3 Sliding window word extraction . . . 195
8.2.4 LSH hyperplanes . . . 197
8.2.5 kNN classification . . . 199
8.2.6 Semantic segmentation of the Defending Yarbus stimuli . . . 202
8.2.7 Runtime of different input lengths . . . 204
8.2.8 Runtime of different number of scan paths . . . 204
8.3.1 User interface ADTF . . . 206

Final remarks
8.3.2 User interface Simulink . . . 207

Appendix
A.1 Evaluation of glint filter . . . 240
A.2 Output example . . . 241
A.3 Classification performance . . . 242
A.4 Evaluation results on the ETRA 2019 challenge dataset . . . 243
A.5 Evaluation results on the Jarbus dataset . . . 244

Tables

Introduction
Structure of this thesis
1.1 Scientific contributions 7

Fundamentals
Retinal processing
2.1 BC convergence 46
2.2 RGC linkage 47

Head-mounted eye-tracking
Circular binary features for robust and real-time pupil center detection
3.1 Decision-based algorithms failure patterns 67
3.2 Runtime evaluation 67
Cycle GANs for pupil and eyelid segmentation
3.3 Segmentation jaccard index 76
3.4 Pupil center detection rate 77
3.5 Pupil center detection rate of refined eye images 77
3.6 Runtime evaluation 78

Remote eye-tracking
Realtime glint detection in remote eye-tracking
4.1 Eye detection rate 91
Evaluation of state-of-the-art pupil detection for remote eye-tracking
4.2 Labeled and detected eye regions 97
4.3 Detection rate 97

Retinal model
Biophysiological model
6.1 Photorecptor parameters 127
Functional model

6.2 Recorded object animation 158
6.3 Detection rate 162

Cognitive eye features
Realtime assessment of cognitive load
7.1 Cognitive load annotations 172
7.2 Cognitive load recognition rates 173
7.3 Runtime evaluation 173
Adaptive eye movement classification
7.4 Classification results 180
7.5 Statistical values of the estimated GMM 181

Appendix
A.1 Hyperparameter overview 239

Abbreviations

Eye physiology

ONL	Outer Nuclear Layer
OPL	Outer Plexiform Layer
INL	Inner Nuclear Layer
IPL	Inner Plexiform Layer
GCL	Ganglion Cell Layer
HC	Horizontal Cell
HC-A	A-Type Horizontal Cell
HC-B	B-Type Horizontal Cell
BC	Bipolar Cell
MB	Midget-Bipolar Cell
FMB	Flat Midget-Bipolar Cell
IMB	Inversive Midget-Bipolar Cell
ON-BC	ON-Bipolar Cell
OFF-BC	OFF-Bipolar Cell
DB	Diffuse-Bipolar Cell
BB	Blue-Bipolar Cell
RB	Rod Bipolar Cell
GBB	Giant Bistratified Bipolar Cell
AC	Amacrine Cell

RGC Retinal Ganglion Cell

MG Midget-Ganglion Cell

PG Parasol-Ganglion Cell

BG Bistratified-Ganglion Cell

WG Wide-Field-Ganglion Cell

PDE Phosphodiesterase

cGMP Cyclic Guanosine Monophosphate

Technology

HMD Head-Mounted Display

CMOS Complementary Metal-Oxide-Semiconductor

NIR Near-Infrared

CAN Controller Area Network

USB Universal Serial Bus

VR Virtual Reality

AR Augmented Reality

IR Infrared

HCI Human-Computer Interaction

HRI Human-Robot Interaction

HUD Head-Up Display

CPU Central Processing Unit

GPU Graphics Processing Unit

VGA Video Graphics Array

ADTF Automotive Data and Time-Triggered Framework

Sensors

LiDAR Light Detection and Ranging

EOG Electrooculography

EEG Electroencephalography

Saliency maps

GBVS Graph-based Visual Saliency

FTS Frequency Tuned Saliency

BMS Boolean Maps Saliency

SWS Spectral Whitening Saliency

Classifier, detectors & machine learning

GMM Gaussian Mixture Model

BMM Bayesian Mixture Model

HMM Hidden Markov Model

k**NN** k-Nearest Neighbor

SVM Support Vector Machine

HC Haar Feature-based Cascade

EM Expectation Maximization

SGD Stochastic Gradient Descent

NN Neural Network

CNN Convolutional Neural Network

GAN Generative Adversarial Network

FC Fully Connected Layer

ReLU Rectifier Linear Unit

ROC Receiver Operating Characteristic

Eye movement classification

I-VT	Velocity-Threshold Identification
I-DT	Dispersion Threshold Identification
I-BDT	Bayesian Mixture Dispersion Threshold Identification
I-VVT	Velocity-Threshold Identification including smooth pursuit
I-VDT	Dispersion Threshold Identification including smooth pursuit
I-PCA	Principal Component Analysis Identification
I-HMM	Hidden Markov Model Identification
I-VMPRay	Rayleigh Test Identification
I-AOI	Area of Interest Identification
MERCY	Moving Estimation Classification

Pupil detection

GRIP	Glint-free Gaze Estimation for Real-time Pervasive head-mounted Eye-tracking
FRED	Fast and Robust Eyelid Recognition
BORE	Boosted-oriented Edge Optimization for Robust, Real-time Remote Pupil Center Detection
PURE	Robust Pupil Detection for Real-time Pervasive Eye-tracking
PUREST	Robust Pupil Tracking for Real-time Pervasive Eye-tracking
ExCuSe	Exclusive Curve Selector
ElSe	Ellipse Selector
SET	Sinusoidal Eye-Tracker
LPW	Labeled Pupils in the Wild

Features & feature selector

FAST Features from Accelerated Segment Test

FAST2 Features from Accelerated Segment Test for Glint Detection

SAX Symbolic Aggregate Approximation

KLT Kanade-Lucas-Tomasi

CBF Circular Binary Features

FCBF Fast Correlation Based Filter

BRIEF Binary Robust Independent Elementary Feature

Methods

SLAM Simultaneous Localization and Mapping

SVD Singular Value Decomposition

ODE Ordinary Differential Equation

CMF Color Matching Function

LSH Local Sensitive Hashing

PCA Principal Component Analysis

RANSAC Random Sample Consensus

Measurements

POR Point of Regard

PCR Pupil-Canthi-Ratio

IPD Interpupillary Distance

ROI Region of Interest

AOI Area of Interest

TTC Time to Collision

REM Rapid Eye Movement

Conferences, journals & institutes

ETRA Eye-tracking Research & Applications

SAP Symposium on Applied Perception

ITSM Intelligent Transportation Systems Magazine

VR Conference on Virtual Reality and 3D User Interfaces

NEUCOMP Journal of Neurocomputing

VISIGRAPP Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications

UBICOMP Conference on Pervasive and Ubiquitous Computing

ICRA International Conference on Robotics and Automation

ICCVW International Conference on Computer Vision Workshops

JOV Journal of Vision

SR Scientific Reports

CFP Call for Papers

QUT Queensland University of Technology

Euro NCAP European New Car Assessment Programme

JEMR Journal of Eye Movement Research

CHAPTER 1

Introduction

Eye-tracking enables real-time sensing and analysis of the user's line of sight. This versatile variable reveals insights into the user's selectivity of visual attention and cognitive situation. In particular the enourmus development of video-based eye-tracking systems in the last decade, pushed the user's line of sight processing and pupil movement analysis to a core modality in HCI, HRI, and psychology research. In video-based eye-tracking cameras record the user's eye and extract significant biometric features, in particular the pupil outline and center. A subsequent software stack processes the eye features towards a Point of Regard (POR) in the scene. Moreover, the fusion of the eye-tracking system with the scene enables the processing of the line of sight in 3D.

However, processing the line of sight is merely the first step, towards comprehensive exploiting of eye-tracking's full potential. Although the user's visual line of sight indicates the focus of the foveal attention, the user's visual perception is by far not limited to the fovea. Novel retinal perception models enable sophisticated prediction of the user's retinal stimulus processing over the whole field of view. Such models are driven by the user's line of sight and parametrize the retinal stimulus processing replication with respect to the retinal perception capabilities in the affected retinal region [2]–[4]. This step beyond the tracked line of sight enables thrilling insights into the user's visual exploration and situation awareness.

Yet, retinal perception is the first step in the human's visual perception pipeline. Complex and highly developed cognitive processes assess and filter the perceived retinal stimulus in terms of significance and semantic relevance. Thus, conscious visual perception is a high-level abstraction of the enormous visual information stream emanating from the retina. Cognitive models, allow approximating this filtering step by predicting, e.g., the performed task, intention, the user's expertise, or cognitive load [1], [8], [11].Eye-tracking allowes to detect such high-level cognitive indicators, and allowes inferences about, e.g., the user's attention bias. Thus, a driver is might more attracted to visual features arising from traffic relevant objects. Vice-versa, a pedestrian, may not pay attention to road signs, even if they are very salient in its field view.

This thesis covers the full pipeline of predicting the user's visual perception, starting

from novel eye-tracking technologies for reliable tracking of the user's line of sight, towards novel and sophisticated retinal processing models, and finally, cognitive models to assess the predicted retinal perception in the user's context.

1.1 Motivation

Modeling and understanding of human visual processing is a long-term core topic in many research areas such as psychology, HCI & HRI, medicine, neuroscience, marketing, and many more. However, human efficiency in selecting fixation targets is highly developed and depends on sophisticated cognitive processing. Driven by the two counteracting processes of preattentive retinal vision and attentive selection, fixation targets in the field of view are continuously assessed in terms of visual and semantic significance. E.g., in human-robot collaboration, such as driving, visual perception is a very limited and time-critical resource in a highly dynamic environment. Real-time prediction of the perceivability of scene objects allows novel methods for advanced assistant systems, e.g., warning of critical but probably unnoticed hazardous traffic situations [8], [21]–[23], minimally invasive attention guidance for navigation and hazard awareness [24]–[26], or evaluation of situation awareness to classify the driver's takeover readiness in highly automated driving [11], [12]. In HCI, Head-Up Displays (HUDs) may assess the perceivability of superimposed objects and could adapt them according to their significance [27], [28]. For example, warning messages are adapted to be as easy as possible to recognize, without covering critical scene contents. Optional information, on the other hand, should be as non-invasive as possible but perceivable, so as to not unnecessarily distract the user from the scene. However, all of these scenarios require a real-time evaluation of the user's visual perception.

Simultaneously, gaze-driven cognitive models utilizing the same sensor – eye-tracking – enable the evaluation of the driver's cognitive situation. By extracting high-level cognitive features, e.g., performed task, intention, the user's expertise, or fatigue, cognitive models reveal a thrilling glimpse into the user's attention bias. Such cognitive features may be derived by pupillometry, fixation and saccade frequencies, or scanpath analysis, enable safe, comfort, and natural human-robot collaboration system, e.g., in conditional automated driving. There, the automated driving system requires to assess the driver's takeover readiness in case the system reaches its limits and needs the driver as a fallback. However, depending on the user's cognitive situation, this handover procedure needs to adapt. Moreover, in combination with retinal perception models, such cognitive models support the estimation of the user's visual situation awareness. Thus, advanced driving assistant systems may evaluate if the driver was able to recognize a potentially hazardous situation. Accordingly, the driver may be supported by an attention guiding, pinpointing its visual attention to the critical traffic scenario. The potential applications of such a joint evaluation of perceived retinal stimulus and cognitive condition are manifold.

1.2 Scope of the thesis

This thesis focuses in analysing the user's gaze beyond the mere mapping of the foveal attention to a scene. Sophisticated models enable joint analysis of the user's cognitive situation, the presented visual scene, and the user's line of sight to predict the user's perceived visual stimulus to an unprecedented degree of detail and accuracy. However, such gaze-driven models rely on accurate and robust eye-tracking technolgy. Thus, the thesis substantiats the state-of-the-art in video-based eye-tracking towards a toughly prediction of the user's line of sight in 3D scenarios. Moreover, this thesis presents two novel models for bio-physiological accurate prediction of retinal perception over the whole user's field view, driven by a tracked line of sight. Inspired by saliency models, these models replicate significant retinal stimulus modulation to predict the scene contents perceivability. Furhtermore, this thesis elaborates cognitive models, based on pupillometry, fixaten and saccade frequencies, and scanpath analysis to provide insights into the user's cognitive conditions. The presented models in this theses allow an extensive analysis of the users visual perception, beyond the tracked line of sight.

The next section introduces the thesis' structure and the author's scientific contributions in eye-tracking research, and gaze-driven perception and cognitive modeling.

1.3 Structure of this thesis & scientific contribution

The thesis is organized into three major parts, prefaced by a comprehensive fundamental chapter. Each part represents a coherent research area and is further subdivided into two core topic chapters. The chapters provide two sections each, elaborating specific scientific contributions of the author. The author's scientific contributions are organized into a method, evaluation, limitation, and conclusion subsection[1]. Figure 1.3.1 outlines the hierarchical structure of the thesis. The thesis starts with the line of sight in Part I. The user's line of sight is measured and inferred using cutting edge eye-tracking technology and provides the core modality for the following Parts II and III. The subsequent Part II processes the user's line of sight in terms of a 3D aligned field of view and evaluates the scene stimulus' perceivability by bio-physiological driven visual perception models. These models provide a bottom-up prediction of the perceivable retinal stimulus. However, although retinal perception is crucial, visual perception is not merely physiological driven. The final substantive Part III elaborates on the human visual perception capabilities and predilections by detecting and incorporating the user's cognitive conditions. Both, Part II and III Together, Part II and III yield in a comprehensive and accurate estimation of a user's visual perception, bio-physiological reasoned, yet incorporating high-level cognitive gaze features. Finally, the thesis is completed by an introduction of the Perception Sandbox, an extensive framework to predict a user's visual perception, developed and patented in the automotive context.

The following briefly summarizes each part's thesis contribution towards a comprehensive

[1] Except Section 5.1, which introduces a scientific toolbox and does not provide a dedicated evaluation section

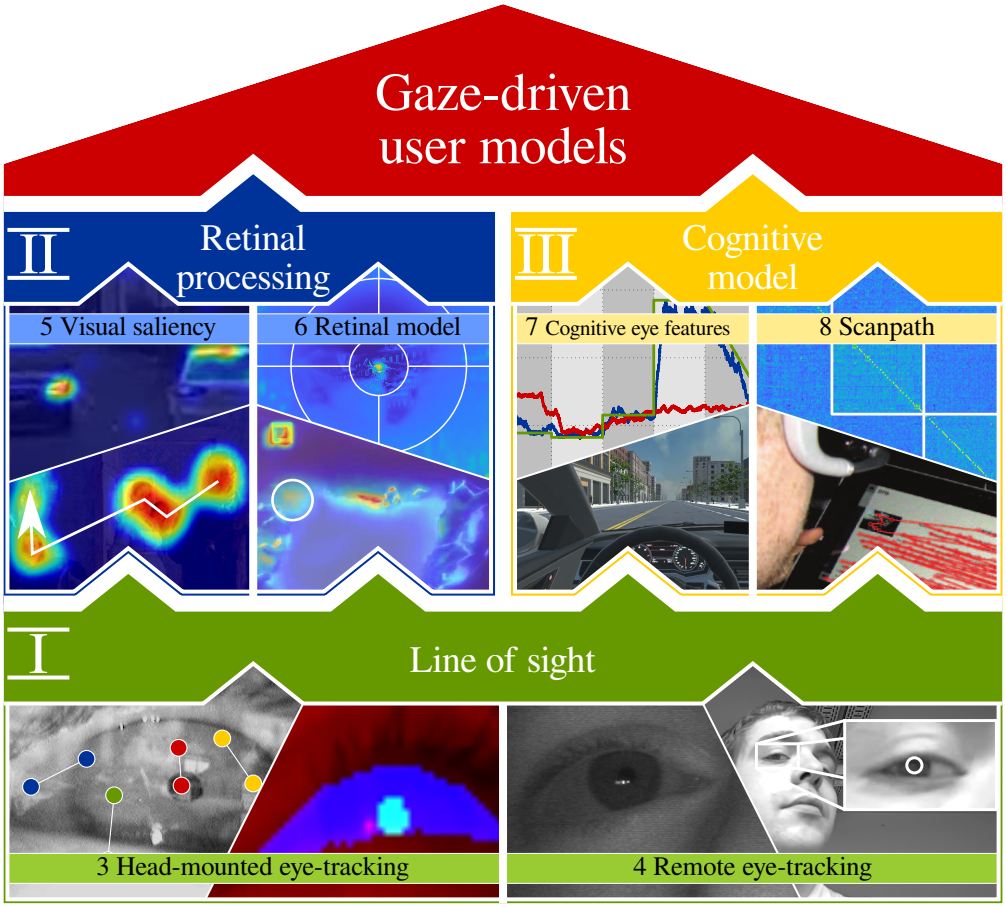


Fig. 1.3.1: Roadmap of this thesis

prediction of a user's visual perception – beyond a tracked line of sight.

Fundamentals The discussed fundamentals in the preface part introduce basic, yet state-of-the-art methods and models to predict the user's line of sight using eye-tracking in the Sections 2.1 to 2.3, retinal perception in Section 2.4, and cognitive conditions in Section 2.5. Each section addresses a particular part in the thesis and conveys essential techniques, methods, and models, as prerequisites in the respective Parts I to III.

Part I Sensing and processing the user's line of sight in a 3D coordinate system is a fundamental step to predict the user's visual perception. The measured line of sight provides the essential parameter to drive retinal and cognitive models. In this context, Part I elaborates two state-of-the-art eye-tracking techniques: video-based head-mounted eye-tracking in Chapter 3 and video-based remote eye-tracking in Chapter 4.

Chapter 3 introduces a novel machine learning-based approach for rapid and accurate pupil detection in head-mounted eye-tracking in Section 3.1 [10]. Moreover, Section 3.2 provides a thrilling deep learning approach to segment, refine, and generate eye images [9]. The subsequent Chapter 4 provides a sophisticated and rapid 3D glint detection method in Section 4.1 [5], followed by an extensive evaluation of state-of-the-art remote eye-tracking pupil detection algorithms in Section 4.2 [13].

Accompanied with the respective fundamental Sections 2.1 to 2.3, this part provides the cutting edge for accurate, yet real-time prediction of the user's 3D line of sight.

Part II Modeling the retinal processing of visual stimuli is a core component to predict the scene's perceivability. Models, simulating the retinal signal modulation, reveal insights about extracted scene features w.r.t. a predicted user's line of sight. This part introduces bottom-up approaches to model the user's bio-physiological visual capabilities and thus provides insights into the visual information flow from the user's retina to the visual cortex.

Hence, Chapter 5 introduces an extensive saliency framework – in Section 5.1 – to detect pronounced and thus likely perceived scene areas [6]. Moreover, Section 5.2 proposes an egocentric adaption of the famous saliency approach Graph-based Visual Saliency (GBVS) to predict a user's retinal stimulus processing w.r.t. a tracked line of sight [2]. The following Chapter 6 introduces the next level of scene perception models by gaze driven evaluation of the user's retinal perception capabilities. These models provide a comprehensive and sophisticated simulation of the retinal signal processing over the whole field of view by incorporating the bio-physiological retinal vision capabilities. Section 6.1 introduces a detailed and most accurate replication of spatial and temporal retinal processing, comprising state-of-the-art findings of human vision research [3]. However, Section 6.2 provides memory and runtime optimized functional derivate of the bio-physiological perception model, to predict the user's visual perception bio-physiological accurate, yet in a real-time fashion [4].

Part III However, consciously visual perception is interwoven to a multitude of cognitive conditions, e.g., intention, task, load, identic memory, and many more. Thus, Part III concludes the significant contribution of the thesis by methods to detect, extract, and evaluate related cognitive features.

Chapter 7 introduces the extraction of low-level, yet significant eye features, i.e., pupil dilation in Section 7.1 [7], and fixations and saccades in Section 7.2 [12], to predict the user's cognitive condition, i.e., cognitive load, fatigue, or performed task. However, Chapter 8 digs deeper into the task classification challenge and provides two state-of-the-art models to analyze the user's temporal visual exploration (scanpath) in terms of eye movements in Section 8.1 [11], and runtime optimized in Section 8.2 [1].

Along with the retinal perception models presented in Part II, Part III provides an essential puzzle piece for a comprehensive evaluation of the user's consciously perceived scene content.

Final remarks The major contributions of the Parts I to III were implemented in a real-time signal processing framework – the perception sandbox – to predict a driver's visual

perception in an online fashion. Moreover, in the year 2019, the acquired findings and technical solutions were transferred into a joint patent of the thesis' author and long term industry partners from Volkswagen AG [20].

Scientific contribution This thesis is supported by 13 significant author's scientific publications as 1st and 2nd authorship from the years 2016 up to 2020. Thus, each thesis part is subsidized by at least four publications published in renowned conferences and journals. Moreover, the thesis' author contributed to 7 further publications, predominantly supporting the fundamental Chapter 2. Table 1.1 provides a full overview of the significant author's publications and which thesis sections they contribute to. However, Table 1.1 only lists the author's 1st and 2nd authorship publications[2].

[2] A full list may found in the bibliography Section 9

Section	Publication
3.1	*CBF: circular binary features for robust and real-time pupil center detection* W. Fuhl, D. Geisler, T. Santini, T. Appel, W. Rosenstiel, and E. Kasneci Eye-tracking Research & Applications (ETRA), ACM, 2018
3.2	*The applicability of cycle GANs for pupil and eyelid segmentation, data generation and image refinement* W. Fuhl, D. Geisler, and E. Kasneci International Conference on Computer Vision Workshops (ICCVW), IEEE, 2019
4.1	*Real-time 3d glint detection in remote eye-tracking based on bayesian inference* D. Geisler, D. Fox, and E. Kasneci International Conference on Robotics and Automation (ICRA), IEEE, 2018
4.2	*Evaluation of state-of-the-art pupil detection algorithms on remote eye images* W. Fuhl, D. Geisler, T. Santini, and E. Kasneci Conference on Pervasive and Ubiquitous Computing (UBICOMP), ACM, 2016
5.1	*Saliency sandbox: Bottom-up saliency framework* D. Geisler, W. Fuhl, T. Santini, and E. Kasneci Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP), SCITEPRESS, 2017
5.2	*Exploiting the GBVS for saliency aware gaze heatmaps* D. Geisler, D. Weber, N. Castner, and E. Kasneci Eye-tracking Research & Applications (ETRA), ACM, 2020
6.1	*A voyage through retinal signal processing: A bio-physiological driven perception model* D. Geisler, P. Berens, and E. Kasneci Scientific Reports (SR) (UNDER REVIEW), Nature, 2020
6.2	*Predicting visual perceivability of scene objects through spatio-temporal modeling of retinal receptive fields* D. Geisler, A. Duchowski, and E. Kasneci Journal of Neurocomputing (NEUCOMP), Elsevier, 2020
7.1	*Person independent, privacy preserving, and real time assessment of cognitive load using eye-tracking in a virtual reality setup* E. Bozkir, D. Geisler, and E. Kasneci Conference on Virtual Reality and 3D User Interfaces (VR), IEEE, 2019
7.1	*Assessment of driver attention during a safety critical situation in VR to generate VR-based training* E. Bozkir, D. Geisler, and E. Kasneci Symposium on Applied Perception (SAP), ACM, 2019
7.2	*On the necessity of adaptive eye movement classification in conditionally automated driving scenarios* C. Braunagel, D. Geisler, W. Stolzmann, W. Rosenstiel, and E. Kasneci Intelligent Transportation Systems Magazine (ITSM), IEEE, 2017
8.1	*Online recognition of driver-activity based on visual scanpath classification* C. Braunagel, D. Geisler, W. Stolzmann, W. Rosenstiel, and E. Kasneci Eye-tracking Research & Applications (ETRA), ACM, 2016
8.2	*A minhash approach for fast scanpath classification* (🏆 Best paper award) D. Geisler, N. Castner, G. Kasneci, and E. Kasneci Eye-tracking Research & Applications (ETRA), ACM, 2020

Part I: Line of sight Part II: Retinal processing Part III: Cognitive model

Tab. 1.1: List of the thesis author's 1st and 2nd authorship publications supporting the thesis Parts I, II, and III.

CHAPTER 2

Fundamentals

This chapter outlines the fundamental concepts of state-of-the-art eye-tracking to predict a user's line of sight, retinal visual processing driven by gaze and stimulus, and finally, the inference of the user's cognitive state to incorporate the task-driven selective attention into the prediction of the user's visual scene perception. The mediated toolset provides the background for Parts I to III – see Figure 1.3.1 – towards a comprehensive prediction of the user's conscious visual perception.

2.1 Eye-tracking

Tracking, recording, and analysis of human eye movements is a long-term core topic in several research areas such as psychology, HRI & HCI, medicine, neuroscience, marketing, and many others. One of the earliest and best-known works in this area is probably the work by A. L. Yarbus, in 1967, who analyzed gaze patterns in watching different complex stimuli [29]. More than 50 years later, thanks to immense technological advances and considerable improvements in terms of handling and availability, eye-tracking has become part of the standard repertoire to analyze human behavior in many tasks.

However, eye-tracking is not limited to the analysis of the user's gaze direction; it also enables inferences about cognitive load [7], [30]–[33], social interaction and cues [34], [35], conscious attention [36], (vision) health [37]–[39], and many more. Hence, eye-tracking boosted to the most promising human-related emerging sensor technology of the last decades. Nowadays, eye-tracking increasingly establishes as an input modality in various HCI systems [40], [41]. Thus, most virtual and augmented reality glasses already offer eye-tracking modules, available either as factory default or opt-in. Next-generation vehicles are equipped with driver monitoring systems – including eye-tracking. Smartphones and tablets use the front camera to detect whether the user is looking on the display – using eye-tracking. Desktop computers already offer standardized interfaces to enable gaze driven applications, from collaborative office applications to gaming.

Yet, in the context of this thesis, eye-tracking is the core sensor to assess the user's line of sight, and thus the orientation of his foveal visual attention. In the following, the three most popular eye-tracking technologies Electrooculography (EOG), scleral-coil tracking,

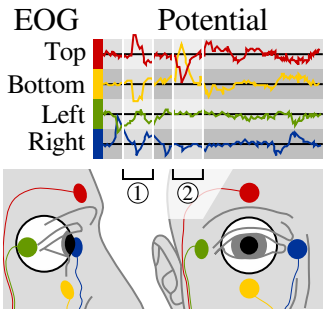


Fig. 2.1.1: Schematic depiction of an Electrooculography (EOG) setup with 4 electrodes: Top (red), Bottom (yellow), Left (green), Right (blue). The electrodes measure the electrical potential at two opposite locations around the eye (Top-Bottom, Left-Right). The highlighted positions in signal ① and ② show typical amplitudes resulting from a diagonal and vertical shift of the eyeball respectively.

and video-based eye-tracking are briefly introduced and reviewed regarding their aptitude to detect a user's line of sight.

Electrooculography

Electrooculography (EOG) uses electrodes attached to the skin around the eye. These electrodes measure shifts as electrical potential, caused by eyeball movements. The technique utilizes the fact that the eyeball behaves like an electric dipole [42]. Thus, eye movements consequently shift the dipole, causing a measurable change in the electrical potential between the attached electrodes. Thus, the spatial distribution of the electrodes around the eye allows to track whether and in which direction the dipole moved. Therefore, electrodes arranged in pairs complementary to each other around the eye. Inverse potential amplitudes between a pair of electrodes indicate a respective dipole rotation. However, using several electrode pairs along distinct axes enables sensing the direction and amplitude of the dipole shift with increasing accuracy.

Figure 2.1.1 illustrates an example of the opposite signal path of complementary electrodes. One of the strengths of EOG, besides the very high sampling rate, is that eye movement can also be observed through closed eyes. However, Figure 2.1.1 also shows that the signal is exposed to noticeable noise. In addition, muscle contractions may significantly overlay the dipole signal. This makes an accurate temporal and spatial identification of the eyeball location very challenging in practice. Furthermore, the required wiring and fixing of the electrodes imposes further limitations for non-laboratory use cases. Therefore, the use of EOG is essentially limited to the analysis of eye movements in laboratory conditions, e.g., to identify the Rapid Eye Movement (REM) sleeping phase [43], or to measure blinks [44], but not suitable to determine the line of sight in everyday scenarios.

Scleral-coil eye-tracking

Similar to EOG, scleral-coils use measurable potential shifts caused by eye movements. However, by attaching a coil built into the contact lens directly to the sclera, the movement is recorded much more accurately, with less interference, and without the need to attach electrodes to the skin.

The scleral-coil is charged by an electric current and thus generates a magnetic field around the contact lens respective pupil. If the magnetic field moves, e.g., by eye movements, a

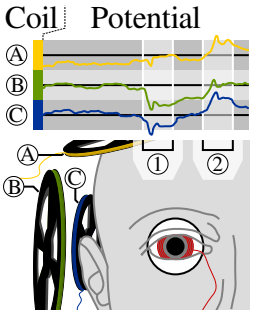


Fig. 2.1.2: Example of a scleral coil tracking setup with the three coils Ⓐ-Ⓒ and the scleral coil in red. The plotted signals show a signal measured in coils Ⓐ-Ⓒ as induced by a horizontal ① and a vertical ② saccade. The first peak results from the induction caused by the eye movement, followed by a depression caused by the current reflux. Subsequently, the signal converges to a level proportional to the eye position.

current is induced in coils arranged around the head, which can be measured analog to EOG electrodes. Moreover, scleral-coils offer the advantage that by using an alternating voltage, the continually changing polarity of the magnetic field can be localized even without eye movements. This enables continuous and accurate measurement of the contact lens location, at very high sampling rates. This makes the scleral-coils the most accurate, yet because of the wired contact lens most inconvenient eye-tracking technique. However, scleral coils are also susceptible to interference, especially with regard to electromagnetic fields. Thus, it must be ensured that no other variable magnetic fields are in the vicinity or are appropriately shielded. Furthermore, the technical challenges for implementation are quite high. While EOG can be realized with standard Electroencephalography (EEG) electrodes, scleral-coils are custom-made and not available on the mass market yet [45], [46].

Figure 2.1.2 shows a scleral-coil setup with three coils around the head. Depending on the direction, eye movements produce a clear pattern in the measured potential of the respective coils. As with EOG, accuracy can be improved by adding additional coils and a versatile arrangement.

Video-based eye-tracking

The broad availability of progressively smaller, but high-resolution video cameras, coupled with ever-increasing computing capacities, led to a veritable boom in video-based eye-tracking in the last decade. Such systems may record eye and scene videos synchronously. An underlying software stack subsequently image-processes and evaluates eye features to predict the user's line of sight towards the scene (video). In a nutshell, one or more cameras are facing towards the user's eyes. The pupil and other significant eye features are extracted from the recorded video stream using state-of-the-art image processing. Subsequently, the extracted features evaluated regarding a user's line of sight, eyelid closure, pupil dilation, and other features. Hence, video-based eye-tracking is very flexible in application and allows to record versatile user parameters that cannot be measured using EOG or scleral-coils.

Consequently, video-based eye-tracking already found its way in a variety of products even outside the laboratory, e.g., Virtual Reality (VR) and Augmented Reality (AR), driver monitoring, or as an additional input modality in smartphones, tablets, and desktop computers [7], [8], [11], [12]. Meanwhile, video-based eye-tracking dominates in the laboratory use cases due to its moderate acquisition costs, its fast and straightforward application, and sufficient accuracy. Besides, due to the low invasiveness, studies benefit from a more natural

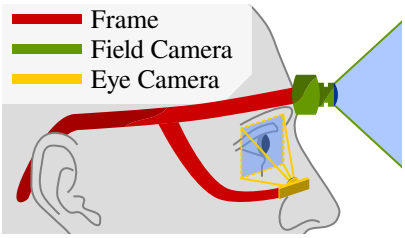


Fig. 2.2.1: Example of a typical video-based head-mounted eye-tracking device. The red structure is a spectacle-like frame. A forward facing camera (green) is mounted on the frame for scene recording. Another camera (yellow) is pointed towards the user's eye to determine the pupil position.

gaze behavior and less technical requirements. Hence, in the following, this thesis refers to video-based eye-tracking.

Video-based eye-tracking divides into two fundamental paradigms: In head-mounted eye-tracking, cameras attached to the user's head extract the line of sight relative to the head pose. Whereas, remote eye-tracking, operates on fixed cameras in the scene and extract the user's line of sight relative to a scene coordinate system. The following Sections 2.2 and 2.3 elaborate both approaches in detail and assesses w.r.t. their applicability to predict a user's scene perception.

2.2 Head-mounted eye-tracking

Video-based, head-mounted eye-tracking systems (also referred to as mobile eye-trackers) enable a user-egocentric localization of the gaze point respective line of sight. A field camera and an eye camera (often one camera for each eye), mounted on a spectacle-like frame record the eye and scene. A subsequent software stack predicts the pupil position in the eye cameras and projects it to the scene. Figure 2.2.1 illustrates a typical setup of a video-based head-mounted eye-tracking device. However, the actual design variates depending on the manufacturer and intended application domain. For instance, many eye-trackers integrated into VR glasses do not require an additional field camera since the scene is already virtual and does not need to be recorded separately [47], [48]. Other eye-tracking systems do not require a field camera, as there is no gaze mapping to the scene envisaged, e.g., for the medical diagnosis of dizziness and balance disorders [49]. Again, other eye-tracking systems use multiple cameras per eye to increase the coverage and resolution of the eye images and even enable stereo processing [50].

However, in the following, this work focuses on head-mounted eye-tracking systems with a dedicated frame (not integrated into VR or AR), providing a field camera and at least one eye camera per eye. These systems are widely used due to their flexible applicability and moderate technical requirements. Besides commercial solutions, there are already numerous published self-developed hardware solutions [51], [52] and freely available software packages [17], [53]–[56]. This makes these systems especially popular for research studies and prototyping, as they can easily be customized and extended to individual requirements. Nevertheless, video-based head-mounted eye-tracking is among the most accurate solutions. Since the sensors are mounted directly on the user, the eyes can be recorded and processed at high resolution.

The following sections provide a comprehensive overview of each aspect in head-mounted

eye-tracking to predict a 3D line of sight in a scene coordinate system. Starting with the recording hardware in Section 2.2.1, to the image processing-based pupil detection in Section 2.2.2, towards a 2D and 3D localization of the gaze point an the line of sight in Sections 2.2.3 and 2.2.4, respectively. The introduced eye-tracking stack focuses on state-of-the-art and publicly available approaches. Moreover, the last two Sections 3.1 and 3.2 push the state-of-the-art in pupil detection and eye image segmentation by two novel machine learning approaches [9], [10]. Both approaches are part of the author's contribution to the research domain of head-mounted eye-tracking systems.

2.2.1 Hardware

The hardware requirements for head-mounted eye-trackers are moderate. Hence, there are already various available construction plans and scientific reports on low-cost self-built eye-trackers. Besides an ergonomic frame, the two most important factors are the sensors, i.e., scene and eye camera.

Eye camera The eye camera is mounted on the frame by a lateral arm, slightly temporal, towards the respective eye. The exact positioning and perspective of the eye camera can be adjusted individually to the user and use case. However, often the camera placement is a compromise between a minimal covert field of view and a reasonable camera perspective of the eye. A high axial displacement to the eye results in less disturbance to the user, since the covered field of view is more in the peripheral. Yet, a steep angle of the camera view to the eye results in inaccuracies in the subsequent visual feature extraction as well as geometric and numerical challenges in calculating the line of sight.

The eye camera is usually an Infrared (IR) sensitive CMOS sensor optimally equipped with a global shutter and a fixed frame rate. Moreover, these sensors usually provide lower resolution but high frame rates. Due to the close placement to the eye and the high sampling rate, even saccades can be reliably recorded and identified. An IR emitter provides constant lighting conditions indoor and enables the application even in dark environments.

Scene camera The field camera is attached to the frame and aligned in the direction of the head. The camera is intended to record the scene from the egocentric perspective of the user. In contrast to the eye camera, the field camera is not in the user's field of view, thus larger sensors with high resolution are installed. Combined with a wide-angle or fisheye lens, large areas of the scene can be captured in high resolution. This capability is especially essential if the scene is evaluated regarding the user's field of view, rather than just the line of sight.

The field camera is usually a commercially and broadly available CMOS RGB camera with an IR light blocker, as used, e.g., in smartphones. In contrast to the eye camera, the field camera typically does not offer high frame rates in favor of a high resolution and bandwidth limitations.

Both cameras are usually connected via USB to a general-purpose computing unit, which provides the image processing, gaze point prediction, and journaling.

2.2.2 Pupil detection

Pupil detection is a crucial step in the video-based eye-tracking stack. Based on image processing, a pupil detection algorithm identifies the geometrical center of the pupil in an eye image. However, depending on the approach, additional features such as the pupil contour, ellipse parameters, or confidence are reported. Meanwhile, there are numerous different pupil detection approaches. The algorithms can be distinguished into two basic approaches: Decision- and machine-learning-based algorithms. However, this section focuses on decision-based algorithms, as they are still predominant in practice. Yet, an examination of machine-learning-based pupil detection found in Sections 3.1 and 3.2.

In the following, the most popular decision-based approaches are briefly introduced, as they were discussed and evaluated in the publications by Fuhl & Geisler et al. in 2016 (Starburst, Świrksi, SET, ExCuSe, ElSe) [13] and more recently from Santini et al. in 2018 (PURE, PUREST, GRIP) [57], [58].

2005 – Starburst is an iterative method, estimating the pupil contour on an input image. It starts at a random point in the edge processed eye image and emits rays in all directions in (like a fan). If a ray exceeds a minimal edge threshold, the related point in the image is marked as a pupil contour candidate. In the second step, all pupil contour candidates emit a burst of reversed rays, back into the direction of the origin, and also marks the first edge the ray exceeds the threshold as a pupil contour candidate. Then, all pupil contour candidates are used to find the best fitting ellipse. This may include several candidates who are not related to the actual pupil contour. Hence, Starburst uses RANSAC[1] for ellipse fitting to remove outliers. The center point of the fitted ellipse is reported as the iteration results and gives the next starting point for the subsequent iteration. If the result converges after some iterations, the last center point is reported as the final detected pupil center [59].

2012 – Swirski et al. estimate the pupil contour in three steps. In the first step, haar-like features are used to preserve the rough pupil region in the input image, and thus reduce the search space. The second step performs a pupil segmentation using k-mean clustering of the pupil area histogram, succeeded by canny based edge detection. Finally, a RANSAC[1] ellipse fitting estimates the pupil contour, w.r.t. the edge gradient direction. [60].

2015 – Sinusoidal Eye-Tracker (SET) presumes the pupil shape to be almost circular. First, the eye image is binarized by a threshold and subsequently segmented by the associated pixel groups. Segments providing less than a certain number of pixels are skipped as outliers. For the remaining segments, the contour pixels are used for ellipse fitting. The center of the ellipse, which shape is closest to a circle, is reported as the pupil center [61].

[1] Random or strategic skipping of data points to remove outliers.

2015 – Exclusive Curve Selector (ExCuSe) comprises internally two different pupil detection algorithms. Depending on a preliminary histogram analysis, one of the two pupil detection algorithms is activated – the first algorithm bases on a canny filtered edge image, which is subsequently refined by morphological operations. Subsequently, the curvature of the edges is analyzed and filtered. The best-suited edges are finally used for an ellipse fitting. The second algorithm detects a rough pupil position using an angle integral projection function. The rough pupil position is iteratively refined until it converges. During each iteration, rays are radiated in all directions from the last determined starting point. If a ray hits an edge in the canny filtered eye image, the intersection point is considered for the subsequent ellipse fit. The center of the ellipse represents the new starting point for the next iteration (analogous to a Starburst). The last determined ellipse center is reported as the detected pupil center [62].

2015 – Ellipse Selector (ElSe) is the successor to ExCuSe. Analogous to ExCuSe, ElSe comprises two algorithms for pupil center estimation. The first algorithm uses a canny-filtered image and morphological operations to filter pupil-related edges. Like ExCuSe, ElSe selects the best-suited edges using different heuristics, e.g., shape and enclosing intensity. The selected edges are used for an ellipse fit giving the pupil center and the full pupil contour. If the first algorithm fails, advanced blob detection is applied to locate the rough pupil center [63].

2018 – Robust Pupil Detection for Real-time Pervasive Eye-tracking (PURE) uses a canny edge filter to identify possible pupil edges, analog to ExCuSe, and ElSe. Yet, PURE renounces a second fallback algorithm if no suitable edges are available for ellipse fitting, and works exclusively on edges. Therefore, curved edge segments are selected on the premise that they are likely distinct parts of the pupil contour. Subsequently, the selected segments are conditionally combined to construct further contour candidates that represent a pupil outline. For each identified candidate, an ellipse is fitted and evaluated w.r.t. its elliptical aspect ratio, the angular extent of its edges relative to the ellipse, and the ratio of the points of the ellipse outline that support the pupil hypothesis. The fitted pupil with the highest confidence level is finally reported as the detected pupil [57].

2018 – Robust Pupil Tracking for Real-time Pervasive Eye-tracking (PUREST) extends PURE by a tracking approach. If no previous pupil position is known, PURE is used as initial detection. If a previous pupil position is known, two different tracking approaches are followed: The first tracker adapts the pupil center during fixation or slow movements. Therefore, the tracker assesses the alignment between the previously reported pupil contour and new extracted edges in a small margin around the previous contour. The second tracker implements a greedy approach, which combines suitable edge segments to reconstruct the pupil when it significantly moved from its previous detected position, e.g., in saccades [58].

However machine learning-based approaches, e.g., PupilNet [64], its successor PupilNet2 [65], and its runtime optimized derivate PupilNetFast from Fuhl et al., as well as

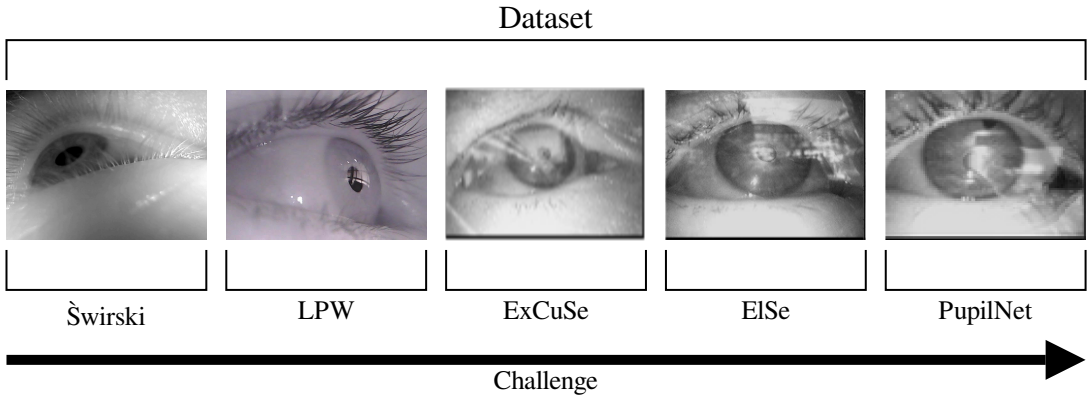


Fig. 2.2.2: Examples images from the datasets used for evaluation. The dataset of Świrski et al. and LPW is considered as a minor challenge, since either no or only moderate reflections occur and the pupil is well visible in the most images. The data sets from Fuhl et al. (ExCuSe, ElSe, and PupilNet) are mainly from driving scenarios involving very dynamic lighting conditions leading to intrusive reflections on the eye surface. Furthermore, the Dataset from PupilNet also includes additional challenges, e.g., ocular physiological phenomena which form a dark spot on the iris.

the two approaches from Fuhl & Geisler et al.: CBF [10], and cycle GAN-based eye segmentation [9], are not considered here, yet extensively elaborated in Section 3.1 and 3.2.

2.2.2.1 Evaluation

The following evaluation is aggregated from the publications of Fuhl et al. and Santini et al. [57], [58], [63]. The algorithms presented above were evaluated on several extensive data sets: Świrski (600 images) [60], ExCuSe ($38,000$ images) [62], ElSe ($55,712$ images) [63], Labeled Pupils in the Wild (LPW) ($130,856$ images) [66], and PupilNet ($41,217$ images) [64]. Figure 2.2.2 shows an example of each of the above-mentioned datasets and ranks them to their included challenges.

All data sets contain contiguous eye images from a video stream. During the evaluation, the order of the eye images was kept according to the video stream to enable PUREST to follow its tracking approach. Furthermore, all images were scaled down to a resolution of 384 px $\times$ 288 px to ensure comparability and to improve runtimes, as ElSe and ExCuSe do this internally anyway. The euclidean distance of the detected to the labeled pupil center was used as an error metric. As the manually labeled pupils also possess a certain amount of noise, an error below 7 px (1.46 % of the image diagonal respective maximal error) is considered acceptable.

Fuhl et al. evaluated his approaches ElSe (74.6 % accuracy) and ExCuSe (63.1 % accuracy) together with Świrski (40.6 % accuracy), Starburst (25.9 % accuracy), and SET (17.8 % accuracy) on the data set of Świrksi, ExCuSe, and ElSe. However, Santini et al.

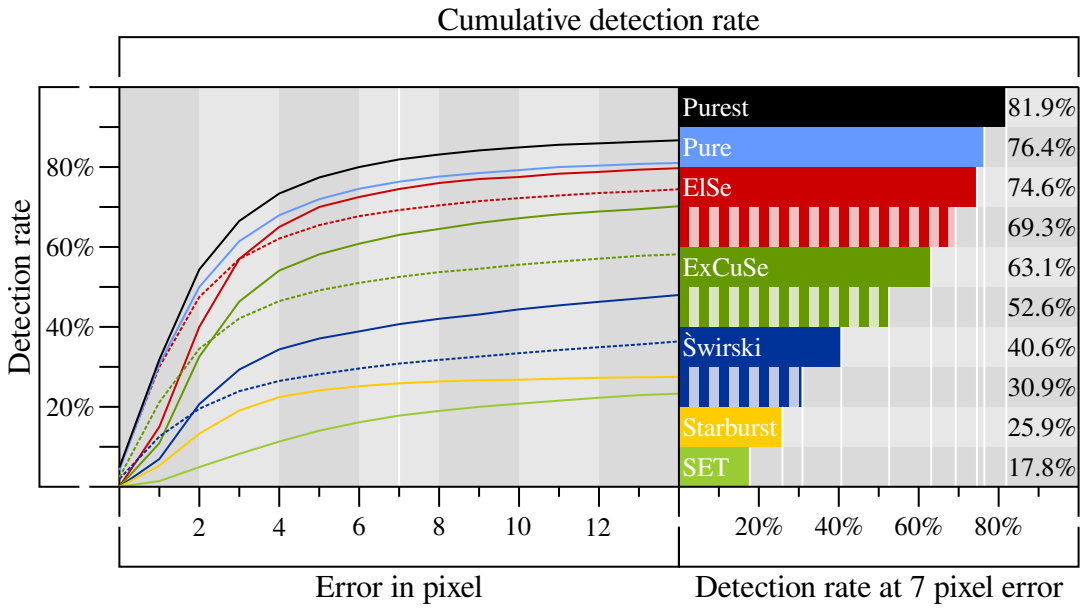


Fig. 2.2.3: Cumulative detection rate of different pupil detection algorithms on eye images from head-mounted eye-tracking. The values are extracted from the publications of Fuhl et al. and Santini et al. [57], [58], [63]. All evaluations were performed on the datasets published with Świrski, ExCuSe, and ElSe [60], [62], [63]. However, the evaluation of PURE and PUREST, as well as the dashed instances of ElSe, ExCuSe, and Świrski also include the LPW, and PupilNet datasets [64], [66].

additionally used the LPW and PupilNet data sets for the evaluation of PUREST (81.9 % accuracy) and PURE (76.4 % accuracy) along with ElSe (69.3 % accuracy), ExCuSe (52.6 % accuracy), and Świrski (30.9 % accuracy).

Figure 2.2.3 shows the evaluation results as a cumulative detection rate. The vertical axis shows the achieved detection rate w.r.t. a given maximal pupil center error on the horizontal axis. The progressive development of video-based pupil detection algorithms in recent decades is noticeable. Starting with Starburst in 2005, the detection rate improved successively over the years with each new proposed algorithm (except SET from 2015), up to the most recent approach Purest from 2018. It is also evident that the tracking approach of PUREST is superior to the increasing performance convergence of mere detection algorithms.

2.2.3 Gaze point

After the successful detection of the pupil center in the eye video, the line of sight is projected to the scene by appropriate mapping function f. In a nutshell, the function f takes the detected pupil parameters as input, e.g., pupil center coordinates, and maps them to a scene coordinate system. For head-mounted eye-tracking, this is, in most cases, the image coordinate system of the scene camera, as shown in Figure 2.2.4. However, speaking about

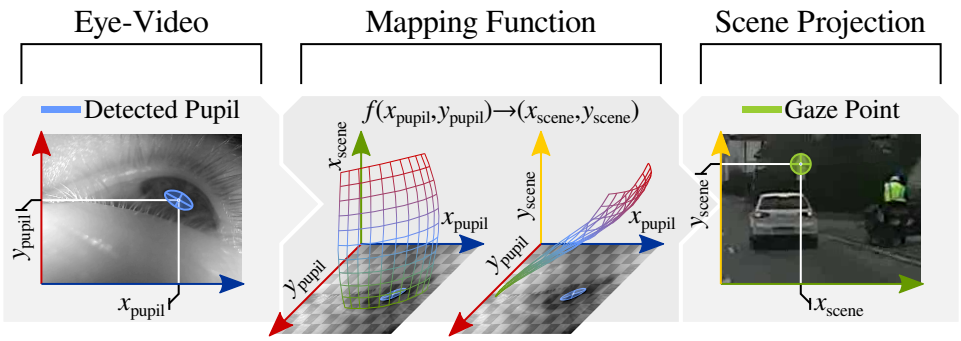


Fig. 2.2.4: A 2D mapping function (center) to project a detected pupil in an eye-video (left) to a gaze point in the scene camera (right).

predicting the line of sight, a projection into a spherical coordinate system is also reasonable. The spherical coordinate system originates in the center of the eye and indicates the direction of the line of sight relative to the head pose. The head pose, in turn, needs to be determined by a further system in order to calculate the line of sight in a 3D scene coordinate system. This representation of the line of sight is needed to locate the field of view in the scene and subsequently estimate the perception of individual scene objects. In the following, different methods to estimate the mapping function f are presented and evaluated.

2.2.3.1 The geometrical challenge

The mapping function f approximates the solution of a series of geometrical problems. Starting with the perspective camera projection of the detected pupil in the eye camera:

$$d_{\text{eye}} \cdot \begin{pmatrix} x_{\text{eye}} \\ y_{\text{eye}} \\ 1 \end{pmatrix} = \underbrace{C_{\text{eye}} \cdot \left(R_{\text{eye}} \middle| T_{\text{eye}}\right)}_{P_{\text{eye}}} \cdot \begin{pmatrix} x_{\text{head}} \\ y_{\text{head}} \\ z_{\text{head}} \\ 1 \end{pmatrix}, \tag{2.1}$$

where $(x_{\text{eye}}, y_{\text{eye}})$ are coordinates in the eye image (e.g., the tracked pupil center). P_{eye} is a 4×3 camera projection consisting of the intrinsic camera parameters C_{eye} (3×3), the camera orientation R_{eye} (3×3), and offset T_{eye} (1×3). $(x_{\text{head}}, \ldots, z_{\text{head}})$ are the respective coordinates in a head coordinate system. d_{eye} is the orthogonal distance between the image plane and the projected point $(x_{\text{head}}, \ldots, z_{\text{head}})$. The same rationale applies to the projection of the scene camera:

$$d_{\text{scene}} \cdot \begin{pmatrix} u_{\text{scene}} \\ v_{\text{scene}} \\ 1 \end{pmatrix} = \underbrace{C_{\text{scene}} \cdot \left(R_{\text{scene}} \middle| T_{\text{scene}}\right)}_{P_{\text{scene}}} \cdot \begin{pmatrix} u_{\text{head}} \\ v_{\text{head}} \\ w_{\text{head}} \\ 1 \end{pmatrix}, \tag{2.2}$$

where $(u_{\text{head}}, \ldots, w_{\text{head}})$ is the respective scene point to a pupil center $(x_{\text{head}}, \ldots, z_{\text{head}})$.

Considering the eye's rotational center as the origin of the head coordinate system, the two points $(x_{\text{head}}, \dots, z_{\text{head}})$ and $(u_{\text{head}}, \dots, w_{\text{head}})$ lie on a ray (optical axis) radiating from the origin. Therefore, the point $(u_{\text{head}}, \dots, w_{\text{head}})$ can be defined as a multiple of q to the point $(x_{\text{head}}, \dots, z_{\text{head}})$:

$$d_{\text{scene}} \cdot \begin{pmatrix} u_{\text{scene}} \\ v_{\text{scene}} \\ 1 \end{pmatrix} = P_{\text{scene}} \cdot \begin{pmatrix} q & 0 & 0 & 0 \\ 0 & q & 0 & 0 \\ 0 & 0 & q & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_{\text{head}} \\ y_{\text{head}} \\ z_{\text{head}} \\ 1 \end{pmatrix}. \tag{2.3}$$

After solving Equations 2.1 and 2.3 for $(x_{\text{head}}, \dots, z_{\text{head}})$, a subsequent substitution results in the following camera-to-camera transformation:

$$\underbrace{R_{\text{eye}}^{-1} \cdot \left(C_{\text{eye}}^{-1} \middle| -T_{\text{eye}}\right)}_{P_{\text{eye}}^{-1}} \cdot \begin{pmatrix} d_{\text{eye}} \cdot x \\ d_{\text{eye}} \cdot y \\ d_{\text{eye}} \cdot 1 \\ 1 \end{pmatrix} = q \cdot \underbrace{R_{\text{scene}}^{-1} \cdot \left(C_{\text{scene}}^{-1} \middle| -T_{\text{scene}}\right)}_{P_{\text{scene}}^{-1}} \cdot \begin{pmatrix} d_{\text{scene}} \cdot u \\ d_{\text{scene}} \cdot v \\ d_{\text{scene}} \cdot 1 \\ 1 \end{pmatrix}, \tag{2.4}$$

where P_{eye}^{-1} and P_{scene}^{-1} are the inverse projections of the eye- and scene-camera, respectively. Using this equation, a detected pupil (x, y, d_{eye}) can be projected to the scene image plane. Therefore, q is solved for each recorded scene pixel (u, v, d_{scene}). The scene pixel with a square error of 0 (or very small in a discrete case) is the intersection of the optical axis in the scene image. Due to the parallax of the scene-camera and the optical axis, this might lead to ambiguous solutions. In this case, the solution with the smallest value for q is selected.

Although the geometric camera to camera projection is certainly elegant, its applicability in practice is rather limited due to the very high technical requirements. Both the eye- and scene-camera must be capable of providing depth images. Moreover, exact calibration of the intrinsic and extrinsic camera parameters is required (including the lens distortion). Finally, there are several further influences which are not addressed in this approach:

1. The **corneal refraction** is caused by the positive meniscus lens-like shape of the cornea on top of the pupil. The consequence is a non-linear displacement of the imaged pupil depending on the camera angle and eyeball rotation, as shown in Figure 2.2.5.

2. The **refraction of glasses** causes similar to the corneal refraction, a non-linear displacement of the imaged eye. However, unlike corneal refraction, it behaves independently to the eyeball's rotation.

3. The **visual axis offset** is the fixed, though individual, offset of the optical axis and the visual axis. While the optical axis describes the line from the center of the rotational center of the eye through the pupil center, the visual axis extends from the foveal center through the pupil center.

Although all these interferences can be modeled geometrically, the number of parameters required to calculate the line of sight increases dramatically. In addition, several parameters, such as the depth image of the eye (d_{eye}) or the refractive index of the cornea or glasses, are

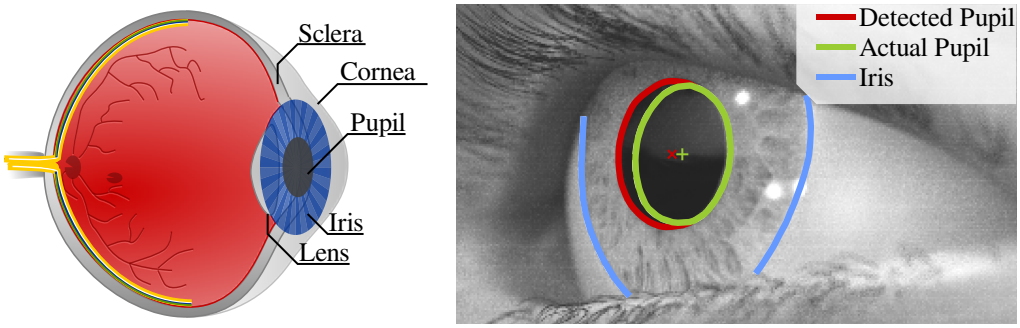


Fig. 2.2.5: Displacement of the depicted pupil due to corneal refraction. As a consequence, the vision based pupil detection algorithm reports a displaced pupil center. In a geometric extrapolation, the incorrect pupil center leads to an offset of the estimated line of sight. Image was adopted from Kuebler et al. [67], [68].

technically complex to measure. Therefore, the practical applicability of a geometric model for eye-tracking is very limited.

However, a closer look at the geometric problem reveals that the deformation of the line of sight is continuous or constant for all three influence types listed above. Thus, it can be assumed that the fundamental shape of the solution domain of Equation 2.4 is distorted, but in principle, follows the shape, as shown in Figure 2.2.4. It turns out that this shape can be successfully approximated by using 2D polynomials.

2.2.3.2 Polynomial Fit

The bivariate polynomial maps the detected pupil center coordinates in the eye video as a gaze point into the scene video. Thereby, the polynomial approximates not only the refraction through the cornea and glasses, as well as the visual axis offset, but also the distance of the pupil to the camera d_{eye}, as shown in Equation 2.4. This is achieved as the pupil moves almost orbital to the rotating eye center. Since the position of the eye camera is static to the rotational eye center, each observed pupil position corresponds uniquely to a distance d_{eye}.

The following equation shows the application of a second-degree polynomial:

$$d_{\text{scene}} \cdot \begin{pmatrix} u_{\text{scene}} \\ v_{\text{scene}} \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} a_{0,0} & \dots & a_{0,5} \\ a_{1,0} & \dots & a_{1,5} \\ a_{2,0} & & a_{2,5} \end{pmatrix}}_{H} \cdot \underbrace{\begin{pmatrix} x_{\text{eye}} \\ y_{\text{eye}} \\ x_{\text{eye}} \cdot y_{\text{eye}} \\ x_{\text{eye}}^2 \\ y_{\text{eye}}^2 \\ 1 \end{pmatrix}}_{\mathcal{P}^2}, \tag{2.5}$$

where d_{scene} is the depth in the scene image. H describes the polynomial constants $a_{0,0}, \dots, a_{2,5}$, and $\mathcal{P}^2$ the polynomial indeterminates of a second-degree polynomial (P^j represents

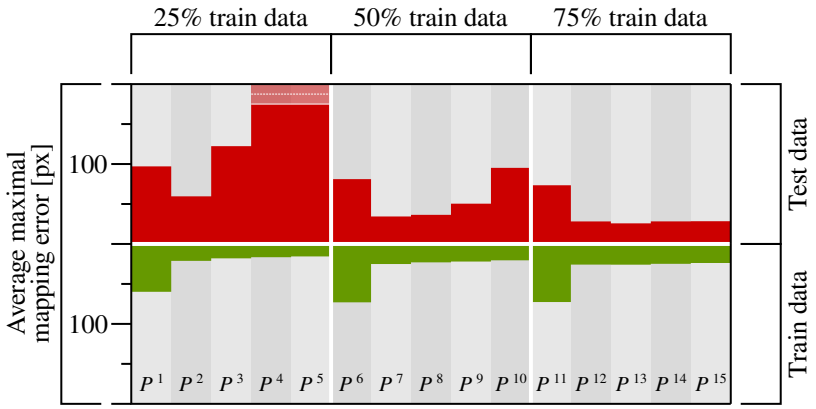


Fig. 2.2.6: Mapping error of the detected pupil by the fitted polynomial. The columns $\mathcal{P}^1$ to $\mathcal{P}^5$ indicate the degree 1 up to 5 of the used polynomial. The three coloumn blocks 25 %, 50 %, or 75 % indicate the proportion of the data used for the fitting (train data). The remaining 75 %, 50 %, or 25 % were used for testing. The bottom row shows the average maximum error of the mapping of the detected pupil center (PUREST) to the scene on the train data. The right column shows the mapping error on the test data. The evaluation dataset was taken from Santini et al.'s CalibMe evaluation (see Section 2.2.3.3) [69].

the indeterminates of the polynomial $\sum_{i=0}^{j} (x+y)^i$). Since d_{scene} is unknown in most head-mounted eye-tracking systems (it would require a field camera with depth perception), d_{scene} is usually assumed to be constant. In this case, Equation 2.5 can be simplified as follows:

$$\begin{pmatrix} u_{\text{scene}} \\ v_{\text{scene}} \end{pmatrix} = \underbrace{\begin{pmatrix} a_{0,0} & & a_{0,5} \\ a_{1,0} & \dots & a_{1,5} \end{pmatrix}}_{H} \cdot \mathcal{P}^2. \tag{2.6}$$

Note that for the linear case $\mathcal{P}^1 = (x, y, 1)^\top$ this corresponds to a 2D homography. This simplification sets the number of unknown parameters to H. H can be determined in a calibration phase. Therefore, the pupil position is recorded together with reference points in the scene image. A common procedure is that a user fixates certain points in the scene. An operator then marks these points in the scene image. The resulting tuples of $(x_{\text{eye}}, y_{\text{eye}})$ eye coordinates and $(u_{\text{scene}}, v_{\text{scene}})$ scene coordinates can then be used to solve the linear system in Equation 2.6 (respective Equation 2.5 if d_{scene} is known) for H.

The strength of the polynomial approach is that all unknowns variables can be estimated in a single calibration session. Moreover, this approach requires no special cameras or additional sensors, e.g., for depth perception. Thus, the polynomial fit can be applied to a variety of systems with moderate hardware requirements. This enables lightweight and flexible eye-tracking systems accommodating even challenging geometric constraints such as glasses or corneal refraction. However, this approach is also subject to some limitations:

1. The **regression error** is caused by the noise in the pupil detection and the mathematical model error. Figure 2.2.6 shows the performance of different fitted polynomials with different coverage of the data points. Therefore, the recorded calibration data points were divided into a 5×5 grid according to their scene image coordinates. Out of this grid, 25 %, 50 %, or 75 % of the cells were randomly selected to fit the polynomials. It shows that higher degree polynomials ($\mathcal{P}^3, \ldots, \mathcal{P}^5$) increasingly tend to overfit at low coverage of the sample points. However, the quadratic polynomial $\mathcal{P}^2$ is quite robust against overfitting even with only 25 % coverage due to its relatively fixed shape. With the increasing coverage of the sample points, higher polynomials show an increasingly smaller fitting error. Nevertheless, due to its robustness at low coverage of data points, and at the same time, similarly good fitting results at higher coverage compared to higher grade polynomials, in most applications, a second-grade polynomial is applied.

2. The **parallax error** is a deliberate modeling error and is caused by the simplified assumption of a constant distance (d_{scene}) between the scene and the user. Consequently, the fitted polynomial is only valid for the scene distance in which the calibration was performed [70].

3. The **susceptibility to extrinsic shifts** is one of the most significant challenges in the daily use and conduction of extensive studies with head-mounted eye-tracking systems. Most head-mounted eye-tracking systems are built to be light as possible for ergonomic reasons. However, the additional weight of the field and eye cameras may cause the frame to slip, especially during longer sessions. But also just talking or grasping the face by the user may cause a movement of the frame. The resulting shift in the extrinsic of the cameras to the eye and scene has fatal implications for the scene mapping [71].

4. The **extrapolation error** results from the indefinite response of the polynomial beyond the calibrated range [72]. In contrast to geometric models, a polynomial regression tends to lead to extreme and unreasonable values outside the interpolated range. Figure 2.2.7 shows the distribution of the regression error. It turns out that especially high errors arise when reference points do not surround the respective area.

Sufficient and well-distributed calibration points can encounter points 1 and 4. Hence, Santini et al. introduced CalibMe in 2017, a procedure that allows even non-experts to quickly and conveniently measure a large number of reference points in a calibration phase [69]. At the same time, CalibMe ensures that the measured eye movements cover a large range of natural eye movements. This ensures that the fitted polynomial works predominantly in the interpolated range. Point 2 can be compensated to a certain extent by a binocular gaze point estimation. However, this requires at least one eye camera per eye. The resulting quadvariate polynomial:

$$\mathcal{P}^n_{\text{binocular}} = \begin{pmatrix} \mathcal{P}^n_{\text{left eye}} \\ \mathcal{P}^n_{\text{right eye}} \end{pmatrix}, \tag{2.7}$$

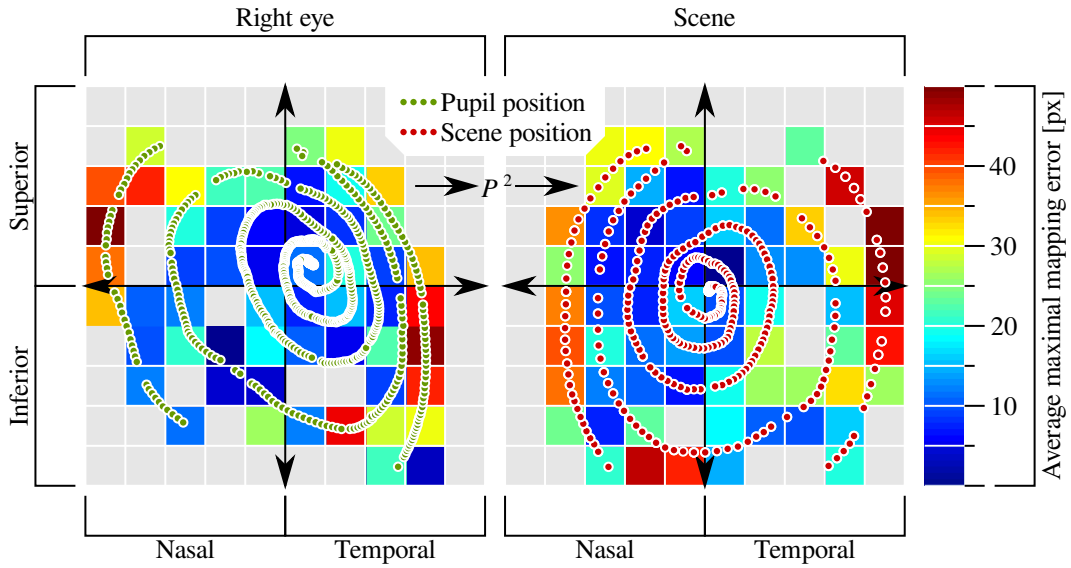


Fig. 2.2.7: The calibration data are divided into a 10×10 grid. For each cell a $\mathcal{P}^2$ polynomial was fitted and evaluated. The data points in the cell were used as test data. All remaining data points were used to fit the polynomial. It demonstrates that especially those instances exhibit a high error that are not surrounded by other data points and thus are extrapolated.

reduces the parallax error due to the incorporated eye vergence [73]. Yet, the more complex model also requires a double amount of calibration points. To address the above point 3, Santini et al. also presented GRIP in 2019 [73]. GRIP uses the outline of the pupil instead of the detected pupil center to perform an extrinsic shift robust mapping of the gaze into the scene camera. All these methods including polynomial fitting are integrated into the eye-tracking platform EyeRec published by Santini and Geisler et al. and are publicly available as open-source [17] on: `https://atreus.informatik.uni-tuebingen.de/santini/EyeRecToo`

2.2.3.3 CalibMe

The classical approach of an eye-tracking calibration is to collect reference points using a n-point calibration scheme. This is usually realized through a planar grid as a calibration pattern, with $n_x \times n_y = n$ corners, e.g., through a printed poster. The calibration pattern is presented to the eye-tracking user, who is instructed to successively gaze at each of the corners, by just moving the eyes (often supported by a chin rest). An instructor or algorithm detects the gazed corner in the field camera and records its position $(u_{\text{scene}}, v_{\text{scene}})$ together with the detected pupil position $(x_{\text{eye}}, y_{\text{eye}})$ as a reference point. In this way, n reference point measurements are recorded, whereby each reference point may consist of several measurements.

This method promises an even sampling of reference points, with a well-controlled density

by n. Yet, in practical application, the n-point calibration poses several challenges:

1. The **number of calibration points** is limited to n. Although n can be selected as desired, the calibration for larger n takes considerably longer and requires more patience from the user.

2. Keeping the head still and focusing a sequence of calibration points by just by eye movements is **counterintuitive**. In natural behavior, the alignment of the line of sight towards a fixation point is a reciprocal process of eye and head movements. A saccade quickly aligns the line of sight to the target, while simultaneously, a slower head movement compensates for extreme eye angles. This intuitive behavior is difficult to suppress even for experienced users. This makes it challenging to record extreme eye angles using a n-point calibration scheme.

3. Using a planar calibration grid inevitably results in **changing distances** $d_{\textbf{scene}}$ to the different calibration points.

CalibMe tackles all these challenges. It uses a single calibration point instead of a fixed scheme of calibration points. The user fixates the calibration point and performs a spiral-like head movement, as illustrated in Figure 2.2.8. Thereby, the user is encouraged to perform the spiral as large and distinct as possible. Thus, the head is maximal turned, while the calibration point is still fixated. Hence, the resulting compensating eye angles, ranging close to the physiological limits. These extreme positions ensure that all natural eye positions are in the interpolated area of the subsequently fitted polynomial. Nevertheless, due to the spiral pattern, a sufficient number of reference points between the extreme angles of the eye are recorded as well. At the same time, through the continuous movement, a large number of different reference points can be acquired. Furthermore, the distance d_{scene} between the eye and the calibration point remains almost constant for each measured reference point. Therefore, the inherent resulting calibrated surface corresponds to a sphere instead of a planar surface, which meets the requirement of Equation 2.6.

As a calibration point, Santini et al. propose to use the center of an ArUCO marker, as it is easy for the user to identify and trace, and simultaneously fast and robustly detectable using state-of-the-art image processing [69], [74]. Therefore, the ArUCO marker is held, similar to a lollipop, by the user with an outstretched arm on a stick. A reference point is sampled whenever the marker is detected in the scene image. This automatization enables the user to perform the calibration without an additional instructor. At the same time, the concept can be easily communicated and demonstrated; thus, even novices can perform a sufficient calibration after a one-time introduction.

2.2.3.4 Glint-free Gaze Estimation for Real-time Pervasive head-mounted Eye-tracking (GRIP)

The polynomial mapping of the pupil center to the scene image suffers, especially when the head-mounted frame with its cameras moves. As a result, the entire extrinsic of the cameras shifts relative to the eye. Thus, the shifted scene position and is not reflected by

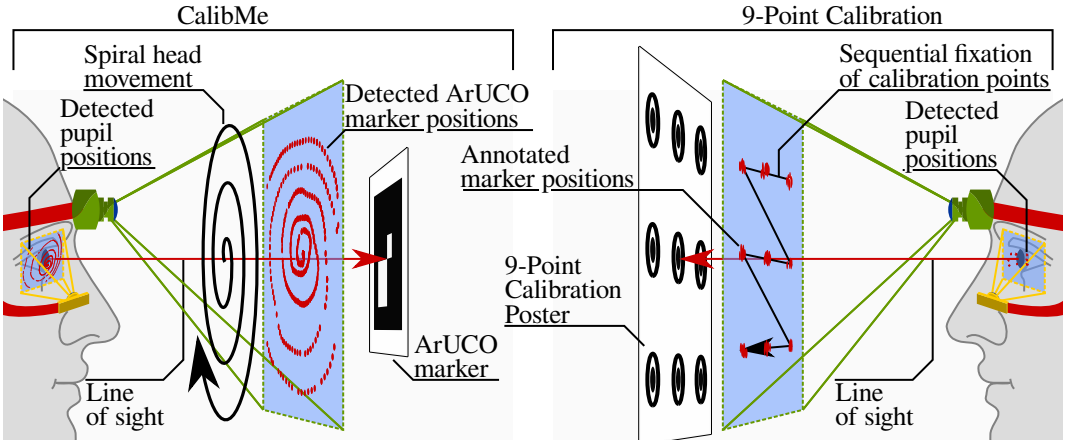


Fig. 2.2.8: Example of a calibration with CalibMe and a 9-point calibration. Using CalibMe, the reference points are collected using an ArUCO marker. The user continuously fixates the centre of the marker and rotates his head in a spiral fashion. However, using the 9-point calibration, the head stays still. The user sequentially fixates the 9 calibration points on the calibration poster by eye movements.

the calibration anymore. In a nutshell, the detected pupil center in the eye camera does not belong to the calibrated point in the scene camera anymore.

Santini et al., therefore, proposed in 2019 a novel method called Glint-free Gaze Estimation for Real-time Pervasive head-mounted Eye-tracking (GRIP), which does not use the pupil center as a feature, but the optical axis as the normal vector v of the circular pupil shape, to map the gaze point into the scene camera [73]. GRIP assumes that although the head-mounted frame may slip as a rigid body, the pose of the cameras relative to each stays unchanged. As a consequence, the optical axis remains the same mapping relationship to the scene camera regardless of the perspective of the eye camera.

However, the major challenge here is to estimate the optical axis relative to the eye camera. Therefore, the detected pupil outline instead of the pupil center is processed, as reported by, e.g., by Pure(st), ElSe, or ExCuSe. Assuming that the pupil is a projected 3D disk, the detected elliptical 2D pupil contour is back-projected into 3D using the circular intersection method [75]. However, this back-projection is ambiguous. Altogether, this leads to four possible solutions of the back-projection: two of them are located on the back of the eyeball and can, therefore, be excluded. Figure 2.2.9 shows the other two possible solutions of the back-projection. The right solution is subsequently selected by a temporal filter. The normal of the selected back-projected pupil disk $v = (\theta_{\text{eye}}, \phi_{\text{eye}}, 1)^\top$ replaces the pupil center image coordinates $(x_{\text{eye}}, y_{\text{eye}})$ as the polynomial input.

Niehorster et al. recently evaluated GRIP, besides three commercial solutions. GRIP can successfully compensate for slippage of the head-mounted frame, by comparable accuracy to the top performers Tobii Pro Glasses 2 and SMI Eye-tracking Glasses 2 [71]. However, the circular intersection method induces a significant amount of noise, which reduces the precision of the raw signal. Yet, post-processing, e.g., using a Kalman filter, may denoise

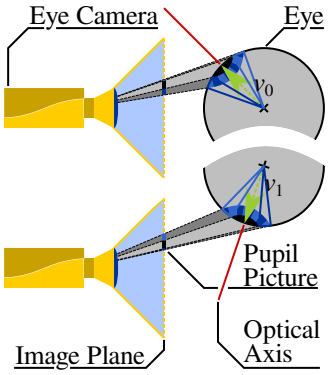


Fig. 2.2.9: Shows the ambiguous solution of the back projection of the pupil from the image plane to 3D using the circular intersection method. Besides the solutions with the normals v_0 and v_1, there are also solutions with the normals $-v_0$ and $-v_1$, on the back of the eyeball. The right normal is selected using a Temporal Filter. The illustartion was adopted and modified from Santini et al. [73].

the signal [76].

Note: Świrski et al. presented a similar approach in 2013, where a 3D model of the eyeball is parametrized by the back-projection of the elliptical outline of the detected pupil [77].

2.2.3.5 EyeRecToo

Santini and Geisler et al. published EyeRecToo in 2017, as a sequel to EyeRec [17], [54]. EyeRecToo is open-source software for real-time pervasive head-mounted eye-tracking. Out of the box, EyeRecToo supports a full eye-tracking stack to run video-based head-mounted eye-trackers:

1. **Image acquisition** and synchronization from USB standard cameras, as used, e.g., by Pupil-Labs, Dikablis, and most customs made eye-trackers.
2. State-of-the-art **pupil detection** and tracking using PURE(st), ElSe, and ExCuSe.
3. Monocular and binocular **gaze mapping** into the scene camera using homography, various polynomials, or GRIP.
4. **Calibration** using the classic n-point calibration or CalibMe.
5. **Image pre- and post-processing**, e.g., undistorting or detection of ArUCO markers.
6. **Storing** and real-time network **streaming** of pupil and gaze information.

Moreover, EyeRecToo is modular structured and easily expandable through various interfaces. Furthermore, the live network data stream and control interface enables EyeRecToo to integrate into complex systems compounds. Due to this flexibility, EyeRecToo has already been adapted and implemented in a wide range of scientific projects. For instance, Braunagel and Geisler et al. used an adapted version of EyeRecToo to stream over a CAN bus interface in a moving based driving simulation environment [11], Bozkir and Geisler et al. adapted EyeRecToo to calibrate and predict the gaze point in an HTC Vive VR [8], or Eivazi et al. integrated EyeRecToo to predict the surgeons gaze in a surgical microscope [78].

However, EyeRecToo is designed to predict the gaze point on a 2D canvas, usually the image plane of a scene camera. The following Section 2.2.4 describes how EyeRecToo

can be extended, in order to use the integrated state-of-the-art pupil detection algorithms, including the slippage robust mapping GRIP, to process a 3D line of sight rather than a 2D gaze point.

2.2.4 Processing the line of sight

To predict how the user perceives a certain scene area, it is crucial to preliminary reconstruct the 3D line of sight. Based on this line, the visual perceptibility of the scene area is evaluated in Part II w.r.t. its eccentricity and appearance. However, this requires an outbreak of the 2D limitation of conventional video-based head-mounted eye-tracking systems. Geisler et al. presented in 2020 a method, in which the line of sight is localized in the scene using multiple ArUCO markers [4]. The ArUCO markers, distributed throughout the scene are fast, robust, and accurately detectable in the scene camera. The detected position of the ArUCO markers in the scene image is subsequently used to localize the scene camera pose $\left(R_{\text{scene}} \middle| T_{\text{scene}}\right)$ in 3D.

This requires at least three detected ArUCO markers, with image coordinates $(x_{\text{ArUCO}\,i}, y_{\text{ArUCO}\,i})$, and their respective scene coordinates $(u_{\text{ArUCO}\,i}, v_{\text{ArUCO}\,i}, w_{\text{ArUCO}\,i})$, to solve the following linear equations system [2]:

$$d_{\text{ArUCO}\,i} \cdot \begin{pmatrix} x_{\text{ArUCO}\,i} \\ y_{\text{ArUCO}\,i} \\ 1 \end{pmatrix} = C_{\text{scene}} \cdot \left(R_{\text{scene}} \middle| T_{\text{scene}}\right) \cdot \begin{pmatrix} u_{\text{ArUCO}\,i} \\ v_{\text{ArUCO}\,i} \\ w_{\text{ArUCO}\,i} \\ 1 \end{pmatrix}, \tag{2.8}$$

where C_{scene} is the previously determined intrinsics of the scene camera. The subscript "ArUCO i" indicates the i-th detected ArUCO marker. $d_{\text{ArUCO}\,i}$ is the orthogonal distance of the ArUCO marker to the field camera image plane. The resulting extrinsic R_{scene} and T_{scene} describe, due to the rigid body of the eye tracker, the pose of the scene camera, as well as the eye cameras (with a constant offset in rotation and translation). Figure 2.2.10 shows a respective setup in a vehicle test-bed. If the exact location of the ArUCO markers is preliminary unknown, it can be estimated in real-time using a Simultaneous Localization and Mapping (SLAM) approach. However, at least the position of one ArUCO must be known or defined as a reference point in the scene coordinate system.

The line of sight is processed as a combination of GRIP, and a polynomial regression of the direction of the line of sight, relative to the rigid body of the head-mounted eye-tracker:

$$d_{\text{los}} \cdot \begin{pmatrix} \phi_{\text{los}} \\ \theta_{\text{los}} \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} a_{0,0} & & a_{0,5} \\ a_{1,0} & \dots & a_{1,5} \\ a_{2,0} & \dots & a_{2,5} \end{pmatrix} \cdot \begin{pmatrix} \phi_{\text{eye}} \\ \theta_{\text{eye}} \\ \phi_{\text{eye}} \cdot \theta_{\text{eye}} \\ \phi_{\text{eye}}^2 \\ \theta_{\text{eye}}^2 \\ 1 \end{pmatrix}}_{\text{GRIP with a } \mathcal{P}^2 \text{ polynomial}}, \tag{2.9}$$

[2] By taking into account the geometry of the ArUCO marker, it is even possible to use a single marker

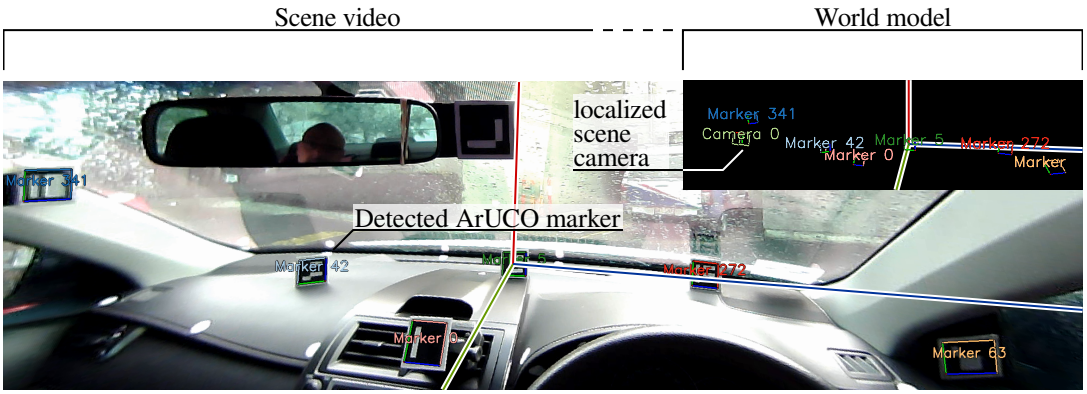


Fig. 2.2.10: ArUCO marker setup in a vehicle test bed of the Queensland University of Technology (QUT), Australia. The camera pose is processed by at least three detected ArUCO marker. The relative position of the ArUCO markers to each other (world model) is continuously updated using UcoSLAM.

where $(\phi_{\text{los}}, \theta_{\text{los}}, 1)^\top$ is the direction of the line of sight. The line equation of the line of sight is defined as follows:

$$\begin{pmatrix} u_{\text{los}} \\ v_{\text{los}} \\ w_{\text{los}} \end{pmatrix} = R_{\text{scene}} \cdot \left(T_{\text{scene}} + T_{\text{offset}} + d_{\text{los}} \cdot \begin{pmatrix} \phi_{\text{los}} \\ \theta_{\text{los}} \\ 1 \end{pmatrix} \right), \tag{2.10}$$

where T_{offset} is the constant offset between the scene camera and the eyeball center (origin of the line of sight). T_{offset} is hard to measure and is, therefore, usually roughly estimated. Since T_{offset} is a constant shift of the visual axis, the influence on the accuracy is moderate. However, the polynomial fit in Equation 2.9 implicitly learns the rotational displacement of the scene camera to the eye (no additional R_{offset} is required). Thus, the fitted polynomial compensates for the translation error of T_{offset} by a rotation. This might lead to an insufficient prediction, especially if the working distance changes significantly compared to the calibration distance. The indeterminate d_{los} indicates the position on the line of sight (note: the direction $(\phi_{\text{los}}, \theta_{\text{los}}, 1)^\top$ is not normalized).

The calibration data can be recorded analogous to CalibMe. Therefore, one of the ArUCO markers in the scene is fixated by the user. During the spiral head movement, the rigid body position $(R_{\text{offset}}|T_{\text{scene}} + T_{\text{offset}})$ of the head-mounted eye-tracker is estimated for each new image of the scene camera according to Equation 2.8, and stored together with the GRIP pupil normal $(\phi_{\text{eye}}, \theta_{\text{eye}})$. Together with the known position of the ArUCO marker, the pose is inserted into Equation 2.10 and resolved to the direction vector:

$$d_{\text{los}} \cdot \begin{pmatrix} \phi_{\text{los}} \\ \theta_{\text{los}} \\ 1 \end{pmatrix} = R_{\text{scene}}^{-1} \cdot \begin{pmatrix} u_{\text{ArUCO}} \\ v_{\text{ArUCO}} \\ w_{\text{ArUCO}} \end{pmatrix} - (T_{\text{scene}} + T_{\text{offset}}). \tag{2.11}$$

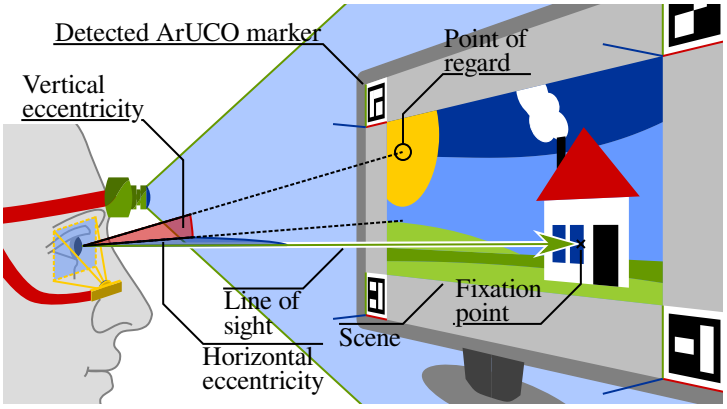


Fig. 2.2.11: Example of a setup to process the line of sight and its eccentricity towards a POR on a desktop screen. Four ArUCO markers enclose the displayed scene and enable the localisation of the scene camera pose. The eccentricity is the angular deviation of the line of sight to the virtual line from the eyeball center to the POR.

$d_{\text{los}} \cdot (\phi_{\text{los}}, \theta_{\text{los}}, 1)^\top$, in turn, is inserted into Equation 2.9 together with the corresponding GRIP pupil normal $(\phi_{\text{eye}}, \theta_{\text{eye}})$ to fit the polynomial coefficients $a_{0,0}, \ldots, a_{n,m}$.

This procedure allows combining the advantages of state-of-the-art pupil detection of, e.g., PUREST, with the slippage robust polynomial mapping GRIP, to determine a 3D line of sight. This line of sight may be used to calculate the eccentricity to the point of regard in the scene. Figure 2.2.11 shows an example of a desktop screen setup, as used in Section 6.2.2.1 to evaluate a retina perception model.

2.2.4.1 Desktop screen setup

The special case of displaying the scene on a desktop screen, as shown in Figure 2.2.11, allows considerable simplification in the calculation of the line of sight. Both the scene on the screen and the image from the scene camera are two planar surfaces. Thus, the 2D gaze point can be transformed from one to the other by using a homography. The homography matrix H can be calculated using the detected ArUCO Makers in the scene camera by solving the following equation:

$$d_{\text{ArUCO}\,i} \cdot \begin{pmatrix} x_{\text{ArUCO}\,i} \\ y_{\text{ArUCO}\,i} \\ 1 \end{pmatrix} = H \cdot \begin{pmatrix} u_{\text{ArUCO}\,i} \\ v_{\text{ArUCO}\,i} \\ 1 \end{pmatrix}, \tag{2.12}$$

where $(x_{\text{ArUCO}\,i}, y_{\text{ArUCO}\,i})$ is the position of the i-th ArUCO marker in the scene camera image, and $(u_{\text{ArUCO}\,i}, v_{\text{ArUCO}\,i})$ the corresponding position in the 2D scene coordinate system. Analogously, the 2D scene camera gaze point $(v_{\text{scene}}, w_{\text{scene}})$ is mapped to the scene screen

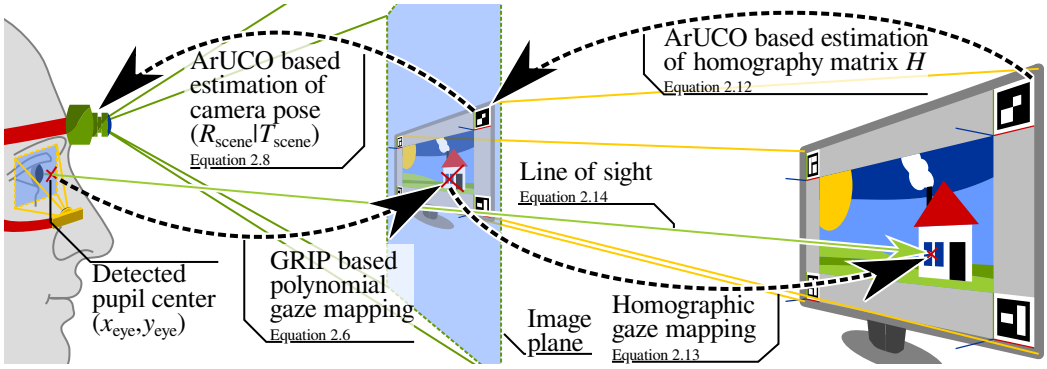


Fig. 2.2.12: Visualization of the straight forward approach to estimate the line of sight in a desktop screen setup. The ArUCO markers are used to calculate the homography H between image plane and desktop screen, as well as to estimate the camera pose respective the eyeball center. GRIP maps the detected pupil center onto the screen plane, which is then projected onto the scene using the homography H. The line of sight results from the estimated eyeball center and the gaze point in the scene.

coordinate system $(v_{\text{screen}}, w_{\text{screen}})$ as follows:

$$d_{\text{screen}} \cdot \begin{pmatrix} u_{\text{screen}} \\ v_{\text{screen}} \\ 1 \end{pmatrix} = H \cdot \begin{pmatrix} u_{\text{scene}} \\ v_{\text{scene}} \\ 1 \end{pmatrix} . \tag{2.13}$$

Furthermore, the pose of the scene camera T_{scene} is calculated according to Equation 2.8, where $w_{\text{ArUCO}\,i} = 0$ and the intrinsic camera parameters of the scene camera C_{scene} are determined according to the screen coordinate system. The line of sight is defined analog to Equation 2.10, but to originate from the center of the field camera T_{scene} and intersect the scene screen at $(v_{\text{screen}}, w_{\text{screen}})$:

$$d_{\text{screen}} \cdot \begin{pmatrix} u_{\text{los}} \\ v_{\text{los}} \\ w_{\text{los}} \end{pmatrix} = T_{\text{scene}} + R_{\text{scene}} \cdot T_{\text{offset}} + d_{\text{los}} \cdot \left(\begin{pmatrix} u_{\text{screen}} \\ v_{\text{screen}} \\ 0 \end{pmatrix} - T_{\text{scene}} \right) . \tag{2.14}$$

2.2.5 Remarks

The previous sections discussed in detail the state-of-the-art in video-based eye-tracking. However, the focus is on the methods used in Parts II and III to calculate a user's line of sight. In fact, the domain of video-based head-mounted eye-tracking is a very dynamic and active research domain. Accordingly, there are a number of further promising approaches that are not elaborated here. In particular, this includes Purkinje-image based methods and machine-learning approaches.

Purkinje-image based methods use reflections of a light source (usually an IR led) to extract further geometric properties of the eye [79]. Theoretically, this enables the geometrical determination of the pose of the eyeball relative to the eye camera. However, most often, the difference between the first Purkinje image (also called glint) and the pupil center is used for a polynomial regression of the viewpoint into the scene camera (PCR) [80]. However, these methods are susceptible to changes in lighting, e.g., walking in sunlight. In addition, these methods must be adapted to the respectively used hardware, in terms of number and arrangement of the glint sources. Nevertheless, this technique plays an essential role in remote eye-tracking and will be elaborated in more detail in the following sections.

So far, machine learning approaches could not prevail despite convincing results. The biggest challenge here is probably a lack of training and evaluation data. These data sets often require manual labeling with enormous time and effort. Nevertheless, Fuhl et al. created the extensive and challenging PupilNet dataset, which was used in conjunction with the ExCuSe dataset (see Figure 2.2.2) to train and evaluate a Convolutional Neural Network (CNN) for pupil detection in 2016: PupilNet [64]. PupilNet already outperforms the state-of-the-art in decision-based algorithms. In 2018, Fuhl and Geisler et al. pushed the bar further ahead with the introduction of Circular Binary Features (CBF) and a random fern based detector. In 2020, Fuhl and Geisler et al. dig deeper into machine-learning-based eye image analysis and published a cycle Generative Adversarial Network (GAN) for a dense segmentation of eye images [9]. Moreover, the introduced cycle GAN architecture can refine eye images in order to improve the detection rates of decision-based pupil detectors significantly. Coupled with the deep learning approaches DeepEye from Vera-Olmos et al. in 2019 [81], DeepVOG from Yiu et al. from 2019 [82], and the real-time CNN from Eivazi et al. in 2019 [83], the recent successful machine learning approaches suggest evidence towards a future, based machine-learning driven image processing in video-based head-mounted eye-tracking.

The following Section 3.1 provides a detailed elaboration to the CBF based pupil detection. Section 3.2, finalizes this chapter by elaborating on the cycle GAN for segmentation, refinement, and generation of eye images.

2.3 Remote eye-tracking

In contrast to head-mounted eye-tracking systems, remote eye-tracking enables non-invasive monitoring of the user's line of sight. Such systems comprise one or multiple stationary cameras to record the user's head and eye movements in a 3D scene coordinate system. The user is unconfined to move within a certain working area limited by the cameras' field of view. Figure 2.3.1 shows the typical setup of a two-camera remote eye-tracking system, as often found to predict the gaze on a desktop screen. The cameras are planary mounted on a rigid frame, aligned towards the user. Additional Near-Infrared (NIR) light sources mounted on the left, and right side provides sufficient illumination for imaging. Moreover, the light source induces characteristic reflections on the eyes' surfaces, so-called Purkinje-images respective glints. Glints are fast and robust detectable and provide additional geometric features [5].

However, the application domain of remote eye-tracking systems is, by far, not limited

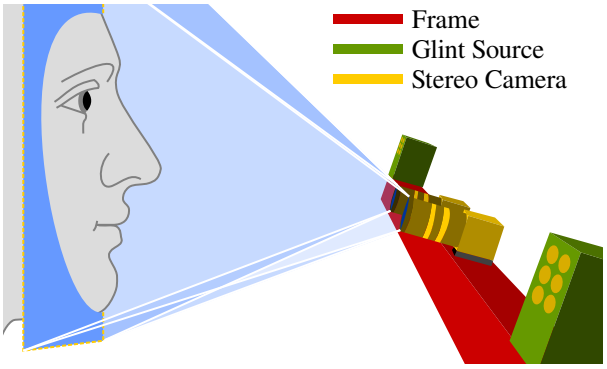


Fig. 2.3.1: Example of a typical remote eye-tracking setup. The green boxes at the bar's corners represent the NIR glint sources, to induce characteristic and fast detectable reflections in the eyes. The two cameras (yellow) mounted on at the center of the bar (red). The stereo configuration enables the 3D detection of glints, pupil center, and other features.

to predict the user's gaze regarding a desktop screen. Much more remote eye-tracking is perfectly suited as an additional input modality in versatile complex HCI systems. If not yet available, the required hardware may seamlessly integrate into existing systems. This increases the user's acceptance and makes remote eye-tracking especially attractive for customer products in various application domains: Advanced driver monitoring in vehicles, smart home, digital assistants, or display based interfaces such as smartphones, laptops, smart displays, and desktop computers. As a contactless and intuitive input modality, remote eye-tracking facilitates natural and convenient interaction between the user and the operating system. Hence, in user analytics and science, remote eye-tracking is one of the most promising emerging sensor technology to assess visual attention, perception, and cognitive load. Non-invasive and transparent, remote eye-tracking enables extensive user studies with minimal side effects and thus natural behavior of the participants.

Yet, in the context of predicting the user's scene perception, remote eye-tracking offers additional advantages. As the eye-tracker is stationary in the scene, the prediction of the line of sight is performed directly in a 3D scene coordinate system. Thus, a transformation from the egocentric 2D perspective to a 3D scene coordinate system as required in head-mounted eye-tracking is not required. However, the distance to the user includes several additional challenges:

1. **Extensive search area**. In contrast to the eye-cameras of head-mounted eye-tracking systems, the cameras are not pointed straight towards the user's eyes. Rather, the cameras cover a fixed working area. Thus, the cameras' perspective to the head and eyes is directly linked to the user's movements. Consequently, in a first step, the remote eye-tracking stack requires to identify the relevant image area of the depicted user.

2. High **off-axial camera perspectives**. As the user is free to move (particularly rotating

its head), independently of the camera's orientation, the eye-tracking system has to deal with high offset angles between the user's face and eyes and the camera. In an extreme case, only a lateral camera perspective is available, resulting in only one or no visible eye in the captured image. Therefore, remote eye-tracking systems must be able to interpret extreme user perspectives while taking into account that relevant user features may not be imaged.

3. **Varying illumination**. Especially in driving scenarios, when sequences of shadows and steep sunlight cause intense light gradients in the image. A challenge that also applies to head-mounted eye-tracking. However, in remote eye-tracking, the dedicated light sources are at more distance and less focused. Accordingly, the supplied illumination hits the user relatively faintly in relation to, e.g., the sunlight. This is particularly challenging if glints are used as reference points.

4. The **low resolution** of the eyes, pupil, and other facial features. Even though the stationary setup usually enables larger camera housings, better optics, and high-resolution sensors, the actual resolution of the relevant image areas is usually significantly lower than in head-mounted eye-tracking. Thus, e.g., the pupil is often depicted on an area of only a few pixels in diameter. Moreover, the whole scene usually involves a larger intensity range as the mere eye region. Hence, the intensity contrast of the eye region just covers a fraction of the camera's pixel depth.

The following sections introduce a straightforward, yet state-of-the-art remote eye-tracking stack, incorporating the aforementioned challenges. First, Section 2.3.1 outlines a typical hardware setup for glint based remote eye-tracking. Followed by an introduction to glint detection in Section 2.3.2, accompanied by a detailed elaboration of a novel and powerful glint detection approach in Section 4.1. The presented glint detection even addresses three of the four named challenges; 1st) the successive reduction and tracking of the search area, 2nd) verification of both eyes visible in the camera image, along with a confidence measure and the user's head pose, and 3rd) robust against various lighting challenges. Section 2.3.3 introduces state-of-the-art remote pupil detection algorithms, specialized for low-resolution and low-contrast eye images. Moreover, Section 4.2 provides an extensive evaluation of pupil detection algorithms, either intended for remote or head-mounted eye-tracking systems. Finally, Section 2.3.4 provides a glint and pupil center based calculation of the user's 3D line of sight.

2.3.1 Hardware

The employed hardware platform of remote eye-tracking systems differs considerably depending on manufacturer, application domain, and subsequent eye-tracking stack. Any combination of mono, stereo, multi-camera setup, using no, one, two, or even more glints, and RGB or near-infrared sensors are found [84], [85]. Moreover, some setups comprise polarizers, hardware synchronization of toggling light sources and camera framerate, additional depth sensors, or even active vision [86], [87]. In fact, the employed hardware is often a trade-off of costs, complexity, and required accuracy. For instance, car manufacturers

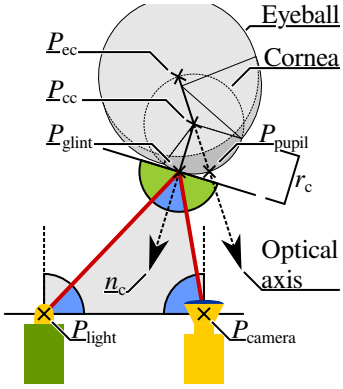


Fig. 2.3.2: Glint based geometrical eye model. The primitive eyeball shape is assumed as spherical like. However, the provided model includes additional spheres to contribute to the bulging sclera, and particularly the cornea. The glint reflects either by the sclera or cornea at the same angle as incidence. The formed triangle of glint source, camera, and glint, enables inferences of the eyes 3D position.

tend to use as few cameras and glint sources as possible, preferably designed as a dedicated and compact unit to reduce cost and complexity. However, in science, the focus shifts more to high accuracy and sampling rates. In the best case, costs and complexity are subordinate factors. Other application domains deal with already available hardware to address a broad audience. E.g., WebGazer requires an arbitrary RGB camera, as found in laptops and smartphones. Thus, enabling quantitative and decentralized studies [84].

The following assumes a hardware design analog to Figure 2.3.1. Two cameras mounted on a rigid frame face towards the user. The cameras are hardware synchronized to enable stereo vision. Furthermore, if several glints are involved, the glints sources alternate each other at the cameras' frame rate. Thus, in each camera is only one identifiable glint-source is active. This design enables a straightforward prediction of the user's line of sight. At the same time, the setup allows deriving more complex systems, e.g., multi-camera or polarization filters, instead of alternating glint sources.

2.3.2 Glints

In theory, glints are robust and fast detectable specular reflections on the user's eye surface (also referred to as the first Purkinje image). The specular reflections, evoked by the humid corneal surface, cause a pronounced point like intensity peak in the camera image. In laboratory conditions, these peaks can be detected and located using simple thresholding methods. Moreover, the position of the glint on a spherical like surface, such as an eyeball, reveals geometric properties in the triangle of glint source, camera, and sphere center, as illustrated in Figure 2.3.2. Both together turn glints into precious features to straightforward predict the eye's position in a remote camera image and reveal insights into the user's eye and head pose.

Glint geometry

The crux in a glint-driven eye model is the known extrinsic of the cameras to the glint's light source. Moreover, the spherical model of the cornea and the eyeball assume the eye's surface as a perfect mirror. Thus, an incident light beam is reflected at the same angle. According to Figure 2.3.2, the center of the corneal sphere P_{cc} can be calculated by the angle bisecting

normal n_c, at a given glint point P_{glint} and corneal radius r_c:

$$P_{cc} = P_{glint} - r_c \cdot n_c, \tag{2.15}$$

whereas n_c depends on the glint's position P_{glint}, the camera's position $P_{camera,i}$, and the glint's light source P_{light}:

$$n_c = \frac{0.5 \cdot P_{light} + 0.5 \cdot P_{camera} - P_{glint}}{\|0.5 \cdot P_{light} + 0.5 \cdot P_{camera} - P_{glint}\|}. \tag{2.16}$$

In a simplified approach, P_{glint} is triangulated in the stereo image and entered into Equation 2.15 and 2.16 to calculate P_{cc}. Yet, depending on the cameras' and light source's position, this may introduce a significant error. The glint point P_{glint} is not an actual scene point, but a specular reflection. Thus, the glint's corneal position depends on the camera's perspective. The error increases with an increasing offset of the cameras to each other. At the same time, a large camera offset is desirable to improve the later pupil triangulation accuracy. To workaround, P_{glint} and n_c must consider as camera-specific:

$$\begin{aligned} P_{cc} &= P_{glint,0} - r_c \cdot n_{c,0}, \\ &= P_{glint,1} - r_c \cdot n_{c,1}, \\ &= \dots, \end{aligned} \tag{2.17}$$

where $P_{glint,i}$ and $n_{c,i}$ correspond to the camera i at position $P_{camera,i}$. Moreover, $P_{glint,i}$ is defined by the respective back-projection of the pixel coordinates $(x_{glint,i}, y_{glint,i})$ and the camera extrinsic $M_{camera,i} = (R_{camera,i}, P_{camera,i})$ and intrinsic $C_{camera,i}$:

$$P_{glint,i} = R^{-1}_{camera,i} \cdot \left(C^{-1}_{camera,i}\middle| - P_{camera,i}\right) \cdot \begin{pmatrix} d_{glint,i} \cdot x_{glint,i} \\ d_{glint,i} \cdot y_{glint,i} \\ d_{glint,i} \\ 1 \end{pmatrix}, \tag{2.18}$$

where $d_{glint,i}$ is the orthogonal distance between the image plane of camera i and the glint point $P_{glint,i}$. Furthermore, $d_{glint,i}$ is assumed as equal across all cameras due to their planar alignment[3]. After inserting Equation 2.18 into the equation system 2.21, the only remaining unknown variable is $d_{glint,i}$. Thus, the equation system 2.21 becomes numerically solvable for at least two cameras.

However, this model only examines corneal glints. To consider also scleral glints, the definition of n_c becomes case dependent. Thus, solving the Equation 2.21 requires either more camera perspectives or multiple glints.

[3]The influence of the corneal curvature on the orthogonal distance is negligible.

Glint detection

A glint based model parametrization requires a robust and accurate glint detection in the remote image. However, this very first step in the remote eye-tracking stack is usually implemented by simple but lightweight image filters, since they have to evaluate the whole remote image:

(Adaptive) Threshold: The majority of existing approaches assume suitable lighting conditions and thus may extract the glints using a fixed threshold [88]–[90]. Yet, some approaches use adaptive thresholds taking into account the average intensity in the closer environment and thus more robust against varying lighting levels [91]. The adaptive threshold is implemented as follows:

$$M_{\text{glint},xy} = \begin{cases} 1 & \text{if } I_{xy} - \hat{c}_{xy} > t_{\text{adaptive threshold}} \\ 0 & \text{otherwise} \end{cases}, \tag{2.19}$$

where $I_{x,y}$ and $\hat{I}_{x,y}$ are the pixel intensities at position (x, y) of the raw and average filtered remote image, respectively. The threshold $t_{\text{adaptive threshold}}$ is the minimum accepted difference between the raw and average filtered intensity. Setting $\hat{I}_{x,y} = 0$ equals to a fixed threshold.

Ebisawa: Ebisawa et al. use first an adaptive threshold as in the above Equation 2.19 to extract temporary glint positions. The position of the glints is determined then more precisely by calculating a gravitational center of the glint candidates within a certain window [92].

Canny, LoG, Sobel: Sharma et al. evaluated the Laplacian of Gaussian (LoG), Sobel, and Canny edge detector with regard to their applicability to glint detection [93]:

$$M_{\text{glint},xy} = \begin{cases} I_{xy} & \text{if edge}_{xy}(I, t_{\text{edge sensitivty}}) \\ 0 & \text{otherwise} \end{cases}, \tag{2.20}$$

where $I_{x,y}$ is the intensity of the input image. $t_{\text{edge sensitivty}}$ is the threshold of the magnitude in order to be counted as an edge (for Canny, it is the upper threshold of the hysteresis).

However, outside laboratory conditions, additional light sources, e.g., sunlight, may induce additional deceptive glints. Furthermore, specular reflecting objects apart from the user's cornea surface, e.g., jewelry, or sunglasses may cause further deceptive reflections and make simple thresholding impossible. Consequently, such threshold-based approaches may lead to many glint candidates and are therefore not applicable outside of confined rooms. Thus, the major challenge in practice is to filter the actual intended induced glint from several deceptive candidates. To address this challenge, Geisler et al. presented in 2018 a sophisticated probabilistic glint candidate filter. Section 4.1 provides an extensive elaboration and evaluation of this approach, along with the new glint detector FAST2.

2.3.3 Pupil detection

The glint's image position provides a fast way to determine the eye region. However, the detection of the pupil in it is an especially challenging task. The low eye image resolution coupled with lower image contrast makes approaches based on edge detection, as predominantly used in head-mounted eye-tracking, unsuitable. Hence, the most state-of-the-art pupil detection algorithms designed for remote eye-tracking rely on image gradients:

Droege & Paulus: Droege and Paulus employ the direction and magnitude of pixel gradients as a feature. This feature reveals to be more stable compared to mere edge-based features for low-resolution eye images. The pixel gradients are threshold-filtered according to their magnitude, as glints or other reflections may cover parts of the pupil. The remaining pixel gradients are interpreted as lines. Subsequently, an M-estimator identifies the joint intersection point of all lines as the coarse pupil center. A generated pupil template further refines the resulting coarse pupil center estimation. The least-squares error of the template identifies the best matching pupil center[4] [94].

Timm & Barth: Analogously to the previous approach, Timm and Barth use the pixel gradient direction as a feature to estimate the pupil center. The underlying assumption is that the gradient angles of the pupil contour must match the iris contour, as both concentric to the pupil center. Therefore, the algorithm iterates over each pixel and calculates its normalized displacement vector to each other pixel (this step is may pre-processed). The squared euclidian distance between the displacement and the gradients of the other pixels is accumulated into a magnitude map. Moreover, Timm and Barth incorporate the inverted pixel intensity to weight the accumulator. Thus dark image areas are emphasized, such as the pupil. The highest amplitude on the magnitude map is the detected pupil center [95].

George & Routray: This approach starts with a coarse pupil center estimation using an orientation annulus convolution filter [96]. Yet, the filter is slightly modified to emphasize vertical edge gradients. The vertical edges of the eyelid may contribute when the eyelids are almost closed, and the lower and upper pupil contours are not visible. Subsequently, a Schaar kernel computes a gradient image, followed by an inverse circular mean filter to emphasize dark regions. The conjunction of both filters results in a likelihood map. Based on the likelihood map, the relationship between each local maxima and the responses is calculated using the standard deviation and the mean value. Based on the calculated relationships, the most likely coarse pupil center is selected. Around the coarse pupil center, all gradients within a certain radius are filtered using a threshold and their deviation to normalized displacement towards the coarse pupil center. The remaining gradients are then medianly filtered and subsequently employed to perform a Random Sample Consensus (RANSAC)-based ellipse fitting[5] [97].

[4] The used implementation uses a least square error approach instead of an M-estimator.

[5] The used implementation implements the least-squares ellipse fitting, as it reveals better empirical results.

Section 4.2 provides a thorough evaluation of the pupil detection algorithms mentioned above on remote eye images, including five candidates from the domain of head-mounted eye-tracking.

The predicted pupil center $(x_{\text{pupil},i}, y_{\text{pupil},i})$ is subsequently triangulated into 3D (P_{pupil} in Figure 2.3.2) by solving the following equation system (e.g., using least square solving):

$$P_{\text{pupil}} = R^{-1}_{\text{camera},i} \cdot \left(C^{-1}_{\text{camera},i}\middle| - P_{\text{camera},i}\right) \cdot \begin{pmatrix} d_{\text{pupil}} \cdot x_{\text{pupil},i} \\ d_{\text{pupil}} \cdot y_{\text{pupil},i} \\ d_{\text{pupil}} 1 \end{pmatrix}, \tag{2.21}$$

over all cameras i (at least two).

2.3.4 Process the line of sight

The optical axis is derived directly from the points 3D P_{cc}, and P_{pupil} calculated in Sections 2.3.2 and 2.3.3, respectively: The straight line through both points. However, the actual line of sight deviates from the optical axis about $5° - 6°$ degrees nasally and $2° - 3°$ degrees superior [98]. The exact deviation varies individually. However, this requires only a one-time one-point calibration for each individual user.

The final line of sight, and thus, the main parameter for the subsequent Part II, is provided as follows:

$$\begin{pmatrix} u_{\text{los}} \\ v_{\text{los}} \\ w_{\text{los}} \end{pmatrix} = P_{\text{ec}} + d_{\text{los}} \cdot (P_{\text{pupil}} - P_{\text{ec}} + O_{\text{los}}), \tag{2.22}$$

where O_{los} is the calibrated offset between the optical axis and the line of sight. P_{ec} is the eyeball's rotational center:

$$P_{\text{ec}} = P_{\text{cc}} - d_{\text{ec,cc}} \cdot (P_{\text{pupil}} - P_{\text{cc}}), \tag{2.23}$$

where $d_{\text{ec,cc}}$ is the distance between the eyeball's rotational center P_{ec} and the corneal center P_{cc}. In the model provided in Figure 2.3.2, $d_{\text{ec,cc}}$ equals r_{c}.

2.3.5 Remarks

This section introduced a straight forward but effective and accurate remote eye-tracking stack to predict a user's line of sight. As aforementioned, the actual number of remote eye-tracking solutions is hardly quantifiable. However, the presented approach provides the fundamental mechanics of glint-based and model-based eye-tracking. In fact, most commercial systems use a variation of the approach presented.

Yet, it should be noted that one approach has is entirely omitted: Pupil-Canthi-Ratio. This approach uses the relationship between corneal glint and pupil position. Based on this relationship, an arbitrary (e.g., polynomial) or model-driven function predicts the user's 2D gaze position on a plane (e.g., screen). For further reading on this approach, the reader is referred to the publications from Zhang et al. [99], [100]. However, as in the context of this

thesis, eye-tracking is used to predict the user's line of sight in 3D, this class of approaches is less relevant.

The accuracy and robustness of the presented approach are directly linked to glint and pupil detection. A profound elaboration of these two challenges is provided in Section 4.1 and 4.2, respectively.

2.4 Retinal processing

Retinal processing is the first step in human vision. How a user perceives the scene depends crucial on how the visual stimulus is processed by the retina and which signals are passed on to the visual cortex via the optic nerve. Yet, the retinal stimulus processing is a sophisticated developed process.

The following Section 2.4.1 and 2.4.2 elaborate two distinct, yet related approaches to analyze the scene's visual content in terms of human perception. First, saliency maps in Section 2.4.1 enable an user independent assessment of the stimulus' visual accessibility. However, the subsequent Section 2.4.2 introduces a bio-physiological driven perspective. This section elaborates on the retinal blueprint and provides fundamental mechanics of human retinal vision. Both sections yield into novel user-oriented perception models, presented in Part II.

2.4.1 Visual saliency

Saliency models predict the significance of a certain scene region raised visual stimulus – however, a stimulus's significance prevalent hinges on the user's context and viewpoint. Thus, a red object's visual appearance may is prominent in a series of distinct colored objects. Yet, the object's color becomes irrelevant if the context requires the user to examine triangle-shaped objects. A more practical related example can found in everyday traffic situations. Prominent colored traffic objects may visually prominent. Yet, from the driver's viewpoint, other features, e.g., the traffic situation, anticipated trajectories, or signals, provide high contextual significance. Thus, drivers consciously focus their attention towards relevant objects respective to the driving task context. Hence, these objects are salient in a specific context but not necessarily due to their visual appearance. However, both factors are crucial in terms of conscious perception. In fact, various statistics report increasing accident rates involving less conspicuous colored vehicles [101]–[103]. Since a colliding vehicle is consistently significant in a driving context, this increase is attributed to the vehicle's low visual saliency.

Hence, Treisman et al. describe conscious perception as a consecutive process of pre- and attentive visual processing. The pre-attentive perception is driven by the retinal visual feature extraction and low-level cognitive processes in the visual cortex. In this stage, the retina continuously assesses the whole field of view in parallel, extracting low-level visual features such as color contrasts, edges, and directional motion (Section 2.4.2 elaborates this process in detail). However, the attentive stage interprets the visual features in intricate patterns, e.g., objects, fore- and background, or movements, and filters according to semantic

relevance [104]. This filtering mechanism is significantly (but not exclusively) driven by higher cognitive processes, e.g., task, intention, and expectations. Thus, the consciously perceived visual stimulus follows a balanced reciprocal process of retinal feature extraction and cognitive filtering.

In the context of saliency models, the two paradigms of feature and semantic driven saliency are referred to as bottom-up and top-down, respectively:

Bottom-up saliency models predict the retinal extracted and emphasized stimulus features. Hence, the visual stimulus is processed by extracting intrinsic scene features followed by a model of the foveal's visual input processing (e.g., ganglions center-surround differences). Moreover, such models may evaluate more complex features, e.g., objectiveness, or fore- and background relations. Yet, bottom-up saliency models do not incorporate the semantic significance of certain scene areas. Thus, these models limited to pre-attentive visual perception. However, as bottom-up models replicate bio-physiological (foveal) retinal and low-level visual cortex processing, they generalize well over distinct application domains. Moreover, bottom-up models may constitute an exaggerated estimation of visual perception, since the subsequent visual processing, interpretation, and consciously representation bases onto the former extracted features.

Top-down saliency models aim to predict visual significance based on semantic relevance. Therefore, usually, a bottom-up salience map is refined by an attentional, task-related, fixation bias [105]–[108]. However, top-down approaches far less homogeneous than bottom-up models, as the mechanics of conscious visual perception are much more complex, individual, and widely less known. In the context of this thesis, top-down saliency models not considered. Moreover, the user's visual attention focus is directly measured as the line of sight. Furthermore, the object's significance is inferred based on the user's performed task and mental condition. E.g., in the driving context, it is assumed that objects reported by the car's environmental sensory systems are relevant to the driving task. Yet, this may include an object's bias based on trajectories, criticality, or type (vehicle, pedestrian, cyclist, etc.), as illustrated in Figure 2.4.1.

The past two decades emerged a whole series of bottom-up saliency models. Since Itti & Koch introduced one of the earliest bottom-up models in 1998, this research domain is extraordinarily active [110]. However, scientific progress evolves into three distinct paradigms:

Static saliency models extract visual features from the spatial domain, e.g., color contrast, edges, fore- and background. A subsequent activation step assesses the features in terms of their visual significance w.r.t. the retinal stimulus processing. However, static models predict the users' average visual perception after a certain time of free viewing of static scenes. Yet, this is not representative of many application domains, e.g., saliency prediction in or from a user's ego perspective.

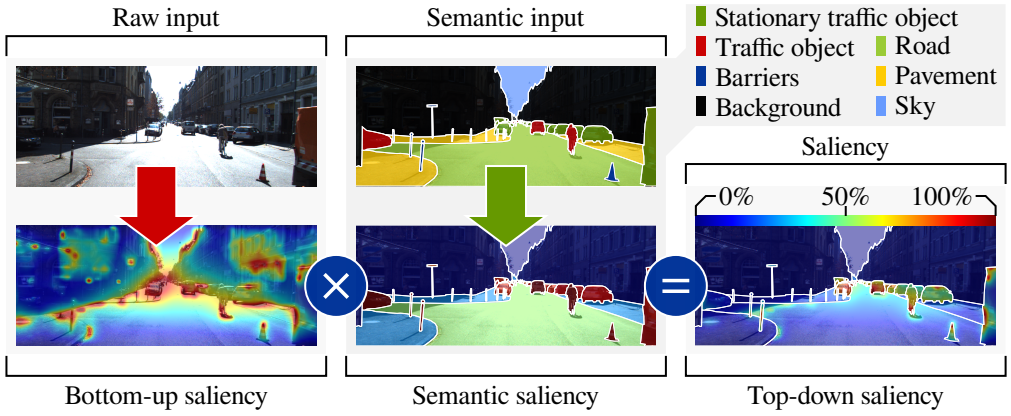


Fig. 2.4.1: Example of a top-down saliency map. The saliency map is processed as a combination of a GBVS bottom-up and semantic segmentation in a driving context. The input stimulus was adapted from the KITTI dataset [109].

Dynamic saliency models address this limitation and leverage static as well as temporal stimulus features. Yet, most dynamic salience models extend static saliency models by motion features, e.g., optical flow, temporal contrasts, or flicker[6] [111]–[114].

Deep-learning saliency models show tantalizing success in modeling deeper cognitive vision processes. Such predominantly CNN-based models may learn besides the mere retinal feature extraction, more abstract relationships, e.g., objectiveness, fore- and background segmentation, or task-specific weighting. More recent deep-learning-based dynamic saliency models incorporate temporal scene dynamics, either as a dedicated input stream or internally extracted temporal features (e.g., through recurrent networks) [115]–[118]. However, these models provide only limited bio-physiological reasoning. Moreover, most deep-learning models are trained and evaluated on huge photography collections, yet known to provide a strong center bias. Furthermore, such large neural networks are still challenging to run in a real-time mode at an appropriate framerate. Hence, this thesis confines to static and dynamic saliency models based on hand-crafted, bio-physiological reasoned visual features.

A comprehensive elaboration of the visual feature extraction along with various state-of-the-art saliency activation algorithms, e.g., Itti & Koch, GBVS, Frequency Tuned Saliency (FTS), etc., is provided in Section 5.1. However, all saliency models predict the average foveal visual stimulus perception over an unspecified time period. Yet, to predict a user's individual visual perception online, it is crucial to 1st) incorporate the line of sight and 2nd) to assess the retinal visual processing, beyond the fovea, over the entire field of view. Section 5.2 presents a straightforward and run-time efficient approach incorporating the user's line of sight into the saliency processing. However, such egocentric saliency models predict the retinal perceptual abilities limited to the foveal stimulus processing. Hence,

[6] Difference of two subsequent frames.

the next Section 2.4.2 provides a deeper glimpse into the retinal bio-physiology and visual processing mechanism. The profound bio-physiological background of the user's retinal stimulus processing across the entire field of view summits in line of sight driven retina models elaborated in Section 2.4.3.

2.4.2 Retinal blueprint

The retina is a hemisphere-shaped structure with a diameter of ∼30 mm to ∼40 mm, providing a highly specialized light-sensitive nerve tissue. Light entering through the pupil is focused and projected by a lens onto the retinal surface. There, photoreceptors absorb photons and propagate the stimulus through a subsequent neural system as an electrochemical signal. The neural system emphasizes and extracts significant visual features and finally transmits them via the optic nerve to the visual cortex. Figure 2.4.2 outlines the basic structure of the retina, along with essential neurons and their interconnections.

The retinal processing of incident stimulus varies significantly according to the affected retinal area. The origin of the line of sight provides the highest density of nerve cells – called the fovea. The fovea has a diameter of ∼5.5 mm, further subdivided into three areas: 1st) The funnel-shaped fovea centralis with a diameter of ∼1.4 mm located in the innermost part provides the highest density of chromatic photoreceptors (cones). Thus, the fovea centralis corresponds to the highest visual acuity. Yet, the fovea centralis exclusively provides photopic vision. 2nd) The surrounding ring of ∼0.5 mm, called parafovea, provides chromatic as well as achromatic photoreceptors and thus provides photopic and scotopic vision. 3rd) The outermost ring of ∼1.5 mm, called perifovea, provide less chromatic, but increasingly more achromatic photoreceptors [119].

Even though the foveal region corresponds to the highest visual capacity, and thus links directly to user's consciously visual attention, the visual perception is by far not limited to it. Much more, the monocular field of view range horizontally from 60° nasal to 107° temporal, and from 70° superior to 80° inferior. Whereas, the binocular field of view extends over ∼114° horizontally.

The visual stimulus is continuously processed across the entire visual field and transmitted via the optic nerve to the visual cortex. The optic nerve exits the retina at ∼3 mm to ∼4 mm nasal. This region is also termed as blind-spot since there are no photoreceptors and, consequently, no visual perception. The blind-spot forms an ellipse with a diameter of ∼1.76 mm horizontally and ∼1.92 mm vertically. Due to the lens' refraction, the blind-spot is located temporally in the user's field of view a ∼18°.

The following sections provide a brief introduction to essential retinal cells and their function in retinal stimulus processing.

2.4.2.1 Photoreceptors

Photoreceptors are light-sensitive sensory cells with cell bodies located in the ONL. These neurons initiate the visual process by hyperpolarizing during light absorption. The human retina has two fundamentally different types of photoreceptor cells: rods and cones (photosensitive retinal ganglion cells not considered). Their pedicles stratify in the OPL, where

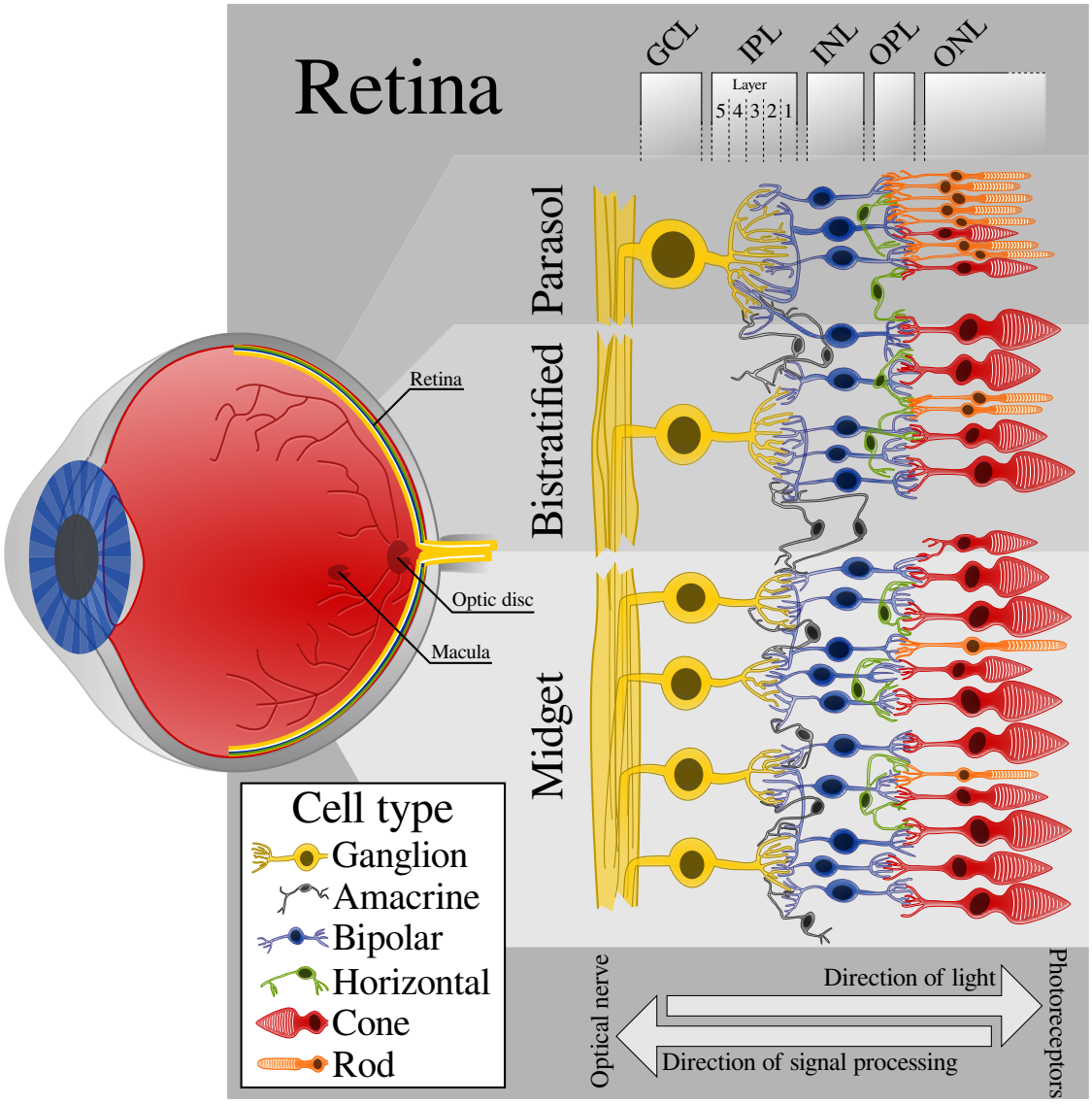


Fig. 2.4.2: Structure of the retina. The human retina is essentially separated into five layers. The Outer Nuclear Layer (ONL) contains the photoreceptor cells (cones and rods), whose axons terminate in the Outer Plexiform Layer (OPL). There, they are linked to the Horizontal Cells (HCs) and Bipolar Cells (BCs) located in the Inner Nuclear Layer (INL). The bipolar cells transfer the signal into the INL, where they are linked to the Amacrine Cells (ACs), also located in the INL, and the Retinal Ganglion Cells (RGCs) located in the Ganglion Cell Layer (GCL). There are primarily three types of RGCs: The Midget-Ganglion Cells (MGs), Bistratified-Ganglion Cells (BGs), and Parasol-Ganglion Cells (PGs).

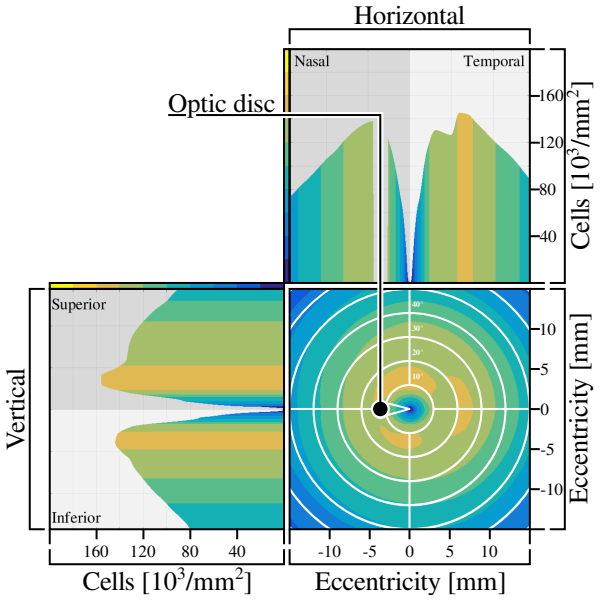


Fig. 2.4.3: Topography of the retinal rods. Rods are located throughout the retina sans for the innermost radius of ∼0.7 mm in the fovea (fovea centralis) and the optical disc. They have the highest density at an eccentricity of ∼3 mm (outside of the fovea). Towards the edges of the retina, the number of rods decreases steadily, while their number decreases rapidly in the direction of the fovea [121], [122].

they link to dendrites of horizontal and bipolar cells (see Figure 2.4.2) [120].

Rods distribute over the entire retina. They reach maximum density at ∼3 mm eccentricity, whereas the innermost ∼0.7 mm of the fovea is entirely free of rods (see Figure 2.4.3 and 2.4.4). The total number of rods significantly exceeds the number of cones (∼91×10^6 rods versus ∼4.5×10^6 cones). Rods show a maximum sensitivity at ∼498 nm, and react all to the same light spectrum. Thus, they do not distinguish between different wavelengths (monochronic vision) and play an essential role in peripheral and night vision.

In contrast to rods, cones have their absorption maxima at different wavelengths and thus, enable color vision. Cones classify into L-, M-, and S-cones. L-cones have maximum sensitivity at ∼564 nm, M-cones at ∼534 nm, and S-cones at ∼420 nm, which corresponds to the colors red, green, and blue, respectively. Cones locate almost exclusively in the foveal area of the retina (see Figure 2.4.5 and 2.4.4) and are much less sensitive than rods; therefore, color perception is limited at night, dusk, or in the periphery. The total number and distribution of S-cones differ significantly from the L- and M-cones. Within a radius of ∼50 µm are no S-cones in the fovea at all, whereas the L/M-cones evince the highest density there. S-cones show the highest density at ∼120 µm to ∼170 µm around the foveal center. The ratio of S- to L/M-cones increases with increasing eccentricity and converges about 7 % (at ∼1.5 mm eccentricity). Up to an eccentricity of ∼4 mm, they constitute only ∼7 % of the total cones population [121]–[123], [125], [126].

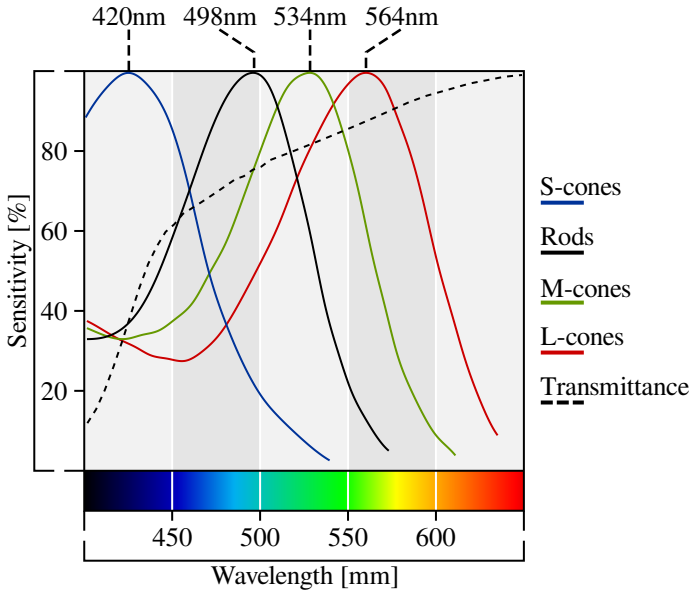


Fig. 2.4.4: Sensitivity of retinal cones and rods. The human effective cone and rod color space. The maximum sensitivity of the S-cones is ∼420 nm (blue), for M-cones ∼534 nm (green) and for L-cones ∼564 nm (red). The maximum sensitivity of the rods is about ∼498 nm [123]. The transmittance of the lens starts at ∼10 % at 400 nm and increases with increasing wavelength up to 100 % ∼650 nm [124].

2.4.2.2 Horizontal Cell

Horizontal Cells (HCs) interconnect multiple photoreceptors laterally and inhibit the signal flow depending on the incidence of light from the surrounding photoreceptors. In this way, they contribute to spatial light adaptation on the one hand and increasing the image contrast on the other. There are at least two types of retinal HCs. A-Type Horizontal Cells (HC-As) are connected exclusively with cones. HC-As are part of the color perception pathway and can be found densely in the foveal area. Whereas, B-Type Horizontal Cells (HC-Bs) interact primarily with rods but also with cones. HC-Bs are part of the luminance pathway, which extends over the entire retina. The dendrite trees of HC-As and HC-Bs range from ∼0.77 mm⌀ and ∼1.04 mm⌀ in the foveal region (≤3 mm eccentricity) to ∼3.64 mm⌀ and ∼6.59 mm⌀ in the peripheral region (at ≥12 mm eccentricity) of the retina [127]–[130].

2.4.2.3 Bipolar Cell

Bipolar Cells (BCs) are located in the INL and transmit the signals of the photoreceptors from the OPL to the Inner Plexiform Layer (IPL). More than ten different types of BCs are known. One of the main traits to categorize BCs is the extent of their axons. The IPL is divided into five strata. BCs, whose axons reach into the 3rd to 5th strata, are referred to as ON-Bipolar Cell (ON-BC), otherwise as OFF-Bipolar Cell (OFF-BC). ON-BCs depolarize

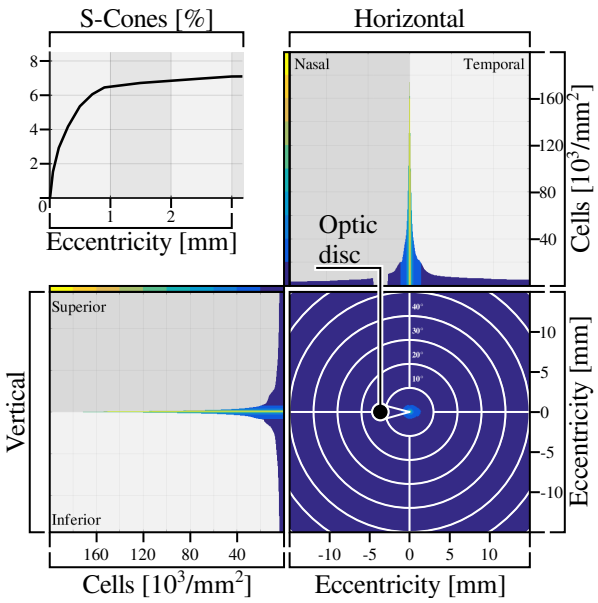


Fig. 2.4.5: Topography of the retinal cones. Cones are strongly concentrated in the foveal area and are sparsely present in the outer foveal, essentially limiting the human color vision to the fovea. There is massive disparity between the total number of L/M-cones and S-cones. Overall, S-cones make up only ∼7 % of the total cone population. Most of them are within a radius of ∼0.12 mm to ∼0.17 mm around the foveal center (perifovea). However, in a inner radius of ∼0.05 mm, no cones are found at all [121], [122].

when the linked photoreceptor is exposed to light, whereas OFF-BCs depolarize vice versa. Thus, they form essential building blocks for the receptive fields of the ganglion cells [131]–[133].

Another BC characteristic is the linked photoreceptor cell type. Midget-Bipolar Cells (MBs) link to a single cone (depending on the size of their dendritic arbor, also termed as Flat Midget-Bipolar Cell (FMB) or Inversive Midget-Bipolar Cell (IMB)), while Diffuse-Bipolar Cells (DBs) converge multiple cones (primary L/M-cones, but also sparsely S-cones). Depending on how far the axons reach into the IPL, DB cells referred to as DB1 to DB6. Blue-Bipolar Cell (BB) converge several S-Cones. The rod's signals transmit via Rod Bipolar Cells (RBs) and Giant Bistratified Bipolar Cells (GBBs). Both linked to several rods, whereby GBBs form giant dendrite trees and probably responsible for the circadian rhythm and not further considered here. Table 2.1 gives an overview of the introduced BC types [131]–[135].

2.4.2.4 Amacrine Cell

Amacrine Cells (ACs) localize in the INL (in some cases, also in the GCL) and link BCs and RGCs laterally in the IPL, similar to HCs in the OPL. Their dendrite arbor usually

	IPL strata					Photoreceptor				
	1	2	3	4	5	converging[1]	L	M	S	R
DB1	■					6.6 ± 1.5	■	■	▒	
FMB	■	■				1.0 ± 0.0	■	■	■	
DB2		■				8.2 ± 1.5	■	■	▒	
DBa		■	■			10.0 ± 1.2	■	■	▒	
DBb		■	■			9.4 ± 1.7	■	■	▒	
DB4			■	■		8.7 ± 0.8	■	■	▒	
DB5				■		7.3 ± 1.1	■	■	▒	
IMB				■	■	1.2 ± 0.4	■	■		
RB				■	■	15 to 30				■
BB					■	1.7 ± 0.8			■	
DB6					■	6.2 ± 1.3	■	■	▒	
	OFF		ON			Mean $\pm$ SD				

[1]The number of converging cones were measured on macaques.

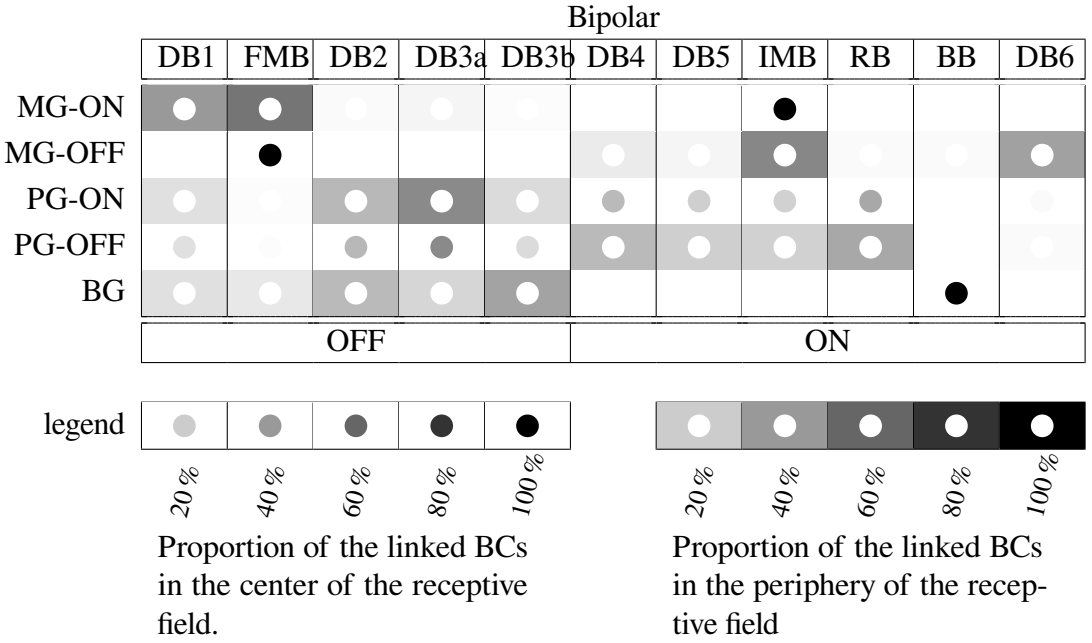
Tab. 2.1: Overview of the different BC types and convergence of photoreceptors. Black fields indicate that the axons of the BC reach into the respective IPL strata, or the corresponding cone is linked to the BC (some S-cones are gray marked since the connections are very sparse) [133], [135], [136].

extends over several strata of the IPL and contact different types of BCs. The function of ACs for signal processing in the IPL appears to be extremely versatile and is still mostly unknown. They form subfunctions in the receptive fields of the RGCs and may play an essential role in the perception of movements and pattern recognition.

More than 30 different AC types are documented, and new types are still being discovered [137]. One of the most important differentiation is the size of their dendritic arbors. A distinction is made between the narrow field ($\sim$0.07 mm$\varnothing$), medium field ($\sim$0.17 mm$\varnothing$), and wide-field ($\sim$0.35 mm$\varnothing$). Furthermore, ACs are often separated using their neurotransmitters (GABA, glycine, or neither). GABA ACs usually belong to the wide-field ACs, the most notable representative of this type is the starburst AC, which plays a role in directionally oriented perception. A well-known representative of the glycinergic AC group is the AII, which takes the signals from several RBs and distribute them to cone BCs, and may play an essential role in scotopic vision [138]–[143].

2.4.2.5 Retinal Ganglion Cell

Retinal Ganglion Cells (RGCs) are located in the GCL of the retina and converge the signals of several BCs and ACs from the IPL and conduct them to the optic nerve. There are likely more than 30 (maybe even up to 80 [144] different types of RGCs, whereby the classification is primarily determined by the extent of their dendritic arbors and the type of the linked BCs and ACs. The three most important, and simultaneously, best characterized



Tab. 2.2: Overview of the RGC's BC linkages. In the fovea, the receptive field of MGs usually consists of only one IMB as the center and one FMB as the periphery, or vice versa. With increasing eccentricity, the number of connected BCs increases, as well as DBs found. As a consequence, the chromaticity of the MGs decreases outside the fovea. PGs have larger dendrite trees and have comparatively many connections to DBs in the center, as well as in the periphery of their receptive fields, which makes them achromatic. PGs, like MGs, can be found with an ON center or an OFF center. In contrast to MGs and PGs, BGs always have an ON center. They form large dendritic trees where they connect BBs at the center and DBs in the periphery of their receptive fields. [133], [135], [136].

are probably the MGs, PGs, and the BGs. All of them form receptive fields by connecting one or more ON-BCs in the center and several OFF-BCs in the periphery, or vice versa. Depending on this, the RGCs are called ON or OFF, based on the type of BCs in the center off their receptive field. Table 2.2 provides an overview of the linked BCs (partly also indirectly via ACs) and their relative proportions in the center or periphery of the receptive field [145]–[147].

MGs generally link a single FMB (MG-OFF) or IMB (MG-ON) in the receptive center. In the foveal region, the periphery is formed in reverse (IMB or FMB). Outside the foveal region, DBs may also be found, as the peripheral complement to the FMB or IMB cells in the center of the receptive field. However, the linkage is selective; thus, they form the color complement of L-cones and M-cones. Recent work shows that in some cases, S-cones are also involved in the receptive center. About 80 % of the RGCs are MG, whereby they accumulate strongly in the foveal area and decrease rapidly with increasing eccentricity. Their dendritic field size range from a single cone in the fovea up to $\sim$0.12 mm$\varnothing$ in the

periphery. Thus, they form the smallest receptive fields in the retina [145], [148]–[150].

PGs form achromatic receptive fields with an ON (PG-ON) or OFF (PG-OFF) center of several bipolar cells, which converge signals from both cones and comparatively many rods. Their dendrites are larger than the MG ($\sim$0.05 mm$\varnothing$ in the foveal and $>$ 0.15 mm$\varnothing$ outside the foveal area) and overlap each other. PGs almost evenly distributed over the entire retina and, therefore, play an important role in peripheral vision. At $\sim$10 %, they are the second most common RGC type [145], [148]–[150].

The receptive fields of the BGs exclusively provide ON center, formed mainly by S-cones (ordinarily by BB) and a diffuse periphery of L/M-cones (yellow). Thus, they form the blue-yellow color complement and are, therefore, termed as chromatic. Their dendrite trees are comparatively huge ($\sim$0.1 mm$\varnothing$ foveal up to $\sim$0.4 mm$\varnothing$ peripheral) and increase by increasing eccentricity, possibly due to the small amount of S-cones. At about 8 % - 10 %, they are much rarer than MGs and, due to the large dendrite trees, also less accurate – resulting in a somewhat blurred perception of blue light. [145], [149], [150].

Figure 2.4.6 shows the topography of RGCs. The highest density is achieved in a band of $\sim$0.5 mm to $\sim$1.5 mm around the foveal center. There are no RGCs in the center, possibly due to space, because of the extremely high number of cones and the funnel-shaped bulge of the retina in this region. A large proportion of the RGCs around the center are, therefore, laterally linked to the cones in the fovea. In fact, the ratio of RGCs to cones is highest in the center of the fovea [151]–[153].

Figure 2.4.7 shows the dendritic field size of different RGC types. The dendritic field tends to grow with increasing eccentricity to the fovea, leading to a decreasing acuity in the periphery.

2.4.3 Retina model

The sophisticated retinal neural structure provides distinct visual perception capabilities depending on the retinal area, the respective cell type population, and their interconnection. Retina models aim to reproduce spatial and temporal retinal stimulus processing to provide a bio-physiological reasoned and accurate prediction of the perceived visual stimulus. However, retinal models divided into two fundamentally distinct approaches:

Bio-physiological models replicate the retinal stimulus processing on a neuron level. Such models simulate electrochemical processes and the associated electric potential modulation on the neuron's interconnection. These approaches are very accurate in the temporal domain and predict the neuron's stimulus processing with astonishing precision. Yet, these models usually reproduce the neuronal interactions of a single retinal functional unit, e.g., the center-surround circuit of a single RGC linked to a couple of BCs and photoreceptors, and are not designed to predict stimulus processing across the entire field of view. Moreover, bio-physiological models not intended to operate on a recorded real-world stimulus. Rather, these models aim to reproduce and model the response of distinct neural circuits as measured in a lab using abstract stimulations, e.g., light pulses [134], [154]–[166].

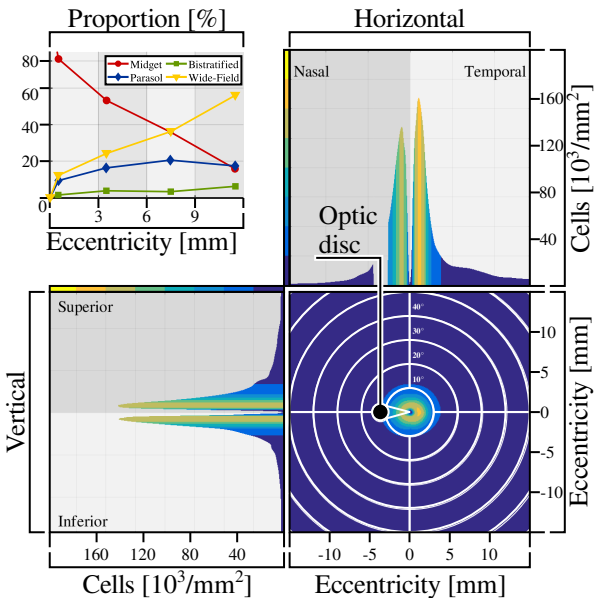


Fig. 2.4.6: Topography of the RGCs. RGCs accumulate strongly in the parafoveal area where they are primarily linked to cones. Their density decreases with increasing eccentricity. In the innermost circle with ∼1.4 mm⌀, the concentration of RGCs decreases considerably. This decrease is possibly because of the funnel-shaped bulge in the retina, which contains a high concentration of cones. The RGCs in the parafoveal area are often laterally connected to the central fovea. MGs are mainly located in the foveal area, which is accompanied by the high density of the cones there. PGs and MGs cells are distributed relatively even over the retina. WGs comprise all RGCs whose size of the receptive fields exceeds that of the BG and are mainly found in perephery [150]–[153].

Functional models , on the other hand, predict the retinal stimulus processing by replicating distinct retinal function unites, e.g., the photoreceptors' color perception, the HCs' contrast adaptation, the BCs' temporal highpass, and signal inversion, up to the RGCs' center-surround difference. Moreover, these models may parametrize the replicated function units based on the stimulus' foveal eccentricity, respective line of sight. The higher abstraction level enables less complex and runtime optimized, yet less accurate, models of retinal signal processing. For instance, the RGCs' center-surround differences of the often implemented as a difference of gaussian, analog to many saliency models. Yet, the Gaussian deviation corresponds to the stimulus' foveal eccentricity to accommodate the increasing dendritic field size in the periphery [167], [168].

However, despite recent achievements, predicting a user's visual stimulus perception remains challenging, not only due to the enormous complexity and capacity of the retinal neuronal processing but also since many retinal biological processes not yet fully understood and thus remain part of active research. This thesis aims to push the state-of-the-art further to a comprehensive model of visual perception. Thus, Section 6.1 introduces a

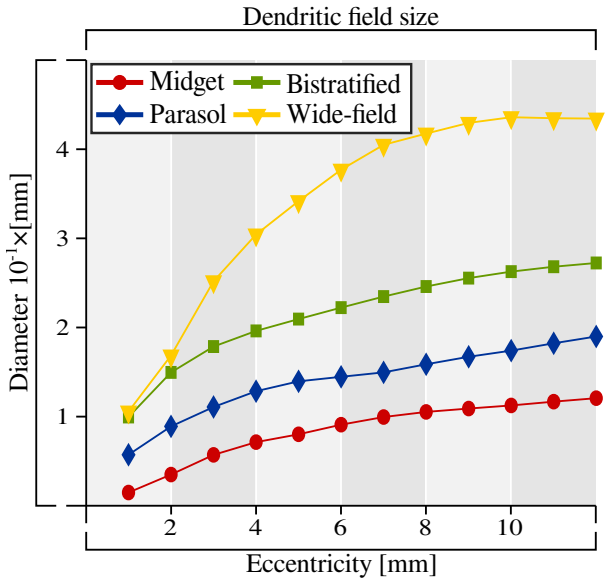


Fig. 2.4.7: Dendritic field size of RGCs. The size of dendrite trees increases with increasing eccentricity for all RGC types. MGs have the smallest dendrite trees of all RGC types. In the fovea centralis, they are often linked to a single cone in the center and in the periphery of the receptive field. The dendritic trees of the PGs are larger than those of the MGs. They are diffusely connected to several cones and rods, both in the center and in the periphery of the receptive field. Their receptive fields are therefore achromatic and have a significantly lower resolution than the MGs. The dendritic trees of the BGs are huge. They are mostly connected to S-cones in the center of their receptive fields and diffuse with L- and M-cones in the periphery. The RGCs termed as wide-field are located in the peripheral and have huge dendrite fields. Their receptive field is usually diffuse in the center and in the periphery [150].

bio-physiological retinal perception model, yet breaking through the limit of state-of-the-art models by incorporating a user's line of sight to parametrize the neural population variety and density along with the stimulus modulation capabilities. The followup Section 6.2 comprises a functional approach, for realtime perception prediction by replicating the most important retinal perceptional function units.

2.5 Cognitive model

Gaze-driven cognitive models aim to predict the user's high-level cognitive conditions, e.g., fatigue, load, expertise, task, and intention. In the context of predicting a user's visual perception, such models enable inferences about the attentive processing of the retinal extracted visual stimulus. For instance, in a driving scenario, detecting a mirror check sequence may provide evidence that the driver intends to change the lane. However, during

the mirror check, an advanced driving assistant system may inferences and incorporates the driver's attentive bias to lane change related traffic objects to predict the driver's visual perception and scene understanding. Vice-versa, detecting a driver's performed secondary task, may indicate an attentive shift towards a non-driving related activity. Thus, a lane change may imply a critical scenario, where an advanced driving assistant requires to support the driver by, e.g., attention guiding or warnings. Such top-down approaches are particularly promising in combination with a retinal processing model as introduced in Section 2.4 and extensively elaborated in Part II.

However, the following sections provide the fundamentals about eye-tracking related cognitive features and assessing techniques, i.e., immediate eye features, such as fixation and saccade metrics, pupillometry, eyelid closure, and more complex, in Section 2.5.1, and more sophisticated features by analyzing the shift of the line of sight in a spatio-temporal domain – the visual scanpath – in Section 2.5.2.

2.5.1 Cognitive eye features

Cognitive eye features describe a set of significant features to predict the user's cognitive conditions immediately derived from the eye image or gaze signal of an eye-tracking system. These features are characterized as particularly straight forward accessible, yet providing a high-level significance, particularly in terms of the user's cognitive situation. Hence, the analysis of these features broad established, e.g., in psychology research.

2.5.1.1 Eye movements

Besides the pupil detection, eye movement identification is one of the most crucial processing steps in an eye-tracking stack. Traditionally, spatial shifts of the pupil detection signal over time are classified into three fundamental types: fixations, saccades, and smooth pursuits.

Fixation Fixations are characterized as a series of detected pupil positions in which the eye apparently is not, or marginally, moving. During this time, the user is focusing the foveal visual attention on a scene point to extract visual information. A typical fixation duration is about 100 ms to 600 ms, yet it may vary significantly depending on, e.g., the user's task, lightning conditions, stimuli complexity, and many more [169].

However, this variation may provide precious insights. Thus, high cognitive load is likely to cause for the same task longer fixation durations, respective lower fixation rates (see also Section 7.1) [7], [8], [30]. Moreover, different tasks may subject to different distributions fixation time, as shown in Section 7.2.

Saccade The saccade links two fixations, and thus describes the process of the pupil's spatial shifts. The visual foveal attention shift is implemented as a rapid ballistic shift of the pupil's position. Therefore, the eyeball may rotate at $30\,^{\circ}\,\mathrm{s}^{-1}$ up to $500\,^{\circ}\,\mathrm{s}^{-1}$. During this fast eye movement, the eye's visual perception is suppressed. Hence, most eye-tracking system filter saccades, even though they are challenging to record (extreme high framerates required). However, the spatial transition between fixated scene points are known to be

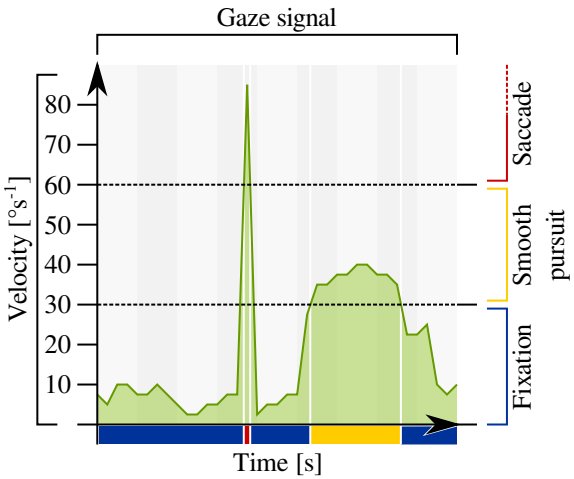


Fig. 2.5.1: Example of a gaze signal's classification using fixed thresholds at $30\,^\circ\,s^{-1}$ for smooth pursuits and $60\,^\circ\,s^{-1}$ for fixates. The velocity classification with fixed thresholds correspondes to the I-VVT. Yet, when using thresholds derived from eye movement's distribution correspondes to I-BDT.

particularly significant, e.g., in terms of performed tasks, intention, and expertise (see also Section 8.1 and 8.2)

Smooth pursuit Smooth pursuits are slow eye movements. They typically emerge whenever the user foveal tracks a scene object, or compensates for ego-motions, e.g., head movements, or driving. The typical range of a smooth pursuit's eye movement is between $10\,^\circ\,s^{-1}$ and $30\,^\circ\,s^{-1}$. However, in this thesis, a smooth pursuit is considered as a peculiarity fixation (as the user fixates a scene object).

Identification The major challenge in identifying eye movement is the eye-tracking system's signal noise. In a perfect world, the eye-tracking system provides a flawless pupil signal in 3D or angular coordinates; thus, eye movements can be straightforward classified according to the mentioned movement intervals. However, in the real world, the eye-tracking system is usually calibrated towards a pixel canvas coordinate system, thus deriving accurate angular eyeball velocities is quite challenging (see also section 2.1). Moreover, sensory and pupil detection noise and limited spatial and temporal resolution raise additional obstacles. Hence, depending on hardware and setup, identifying eye movements is one of the oldest yet still a present challenge.

However, there are plenty of different approaches to classify the pupil signal into fixations and saccades (and sometimes smooth pursuits). Holmqvist and Kennedy et al. provide a comprehensive overview of eye movement classification techniques from 2011 and 2016 [170], [171]. Yet, the following briefly introduces the most popular approaches:

Velocity-Threshold Identification (I-VT): The I-VT is the most straightforward eye movement classification. A fixed threshold separates the eye movement's velocity into fixations and saccades [172]. However, in its original definition, the I-VT does not distinguish between fixations and smooth pursuits. Yet, the I-VT is often extended by an additional threshold, subdividing fixations into smooth pursuits (I-VVT) [173]

Dispersion Threshold Identification (I-DT): Dispersion filters identify the eye movements by the intersample distance within a temporal window. Similar to I-VT, a fixed

threshold separates the eye's dispersion into fixation and saccades [172]. Furthermore, like I-VT, the original I-DT does not separate smooth pursuits, yet can be extended by the same paradigm (I-VDT) [173].

Bayesian Mixture Dispersion Threshold Identification (I-BDT): Both I-VT and I-DT raise the challenge to determine a proper threshold parametrization. However, I-BDT solves this issue through a data-driven approach. By estimating a Gaussian Mixture Model (GMM), the I-BDT implicitly learns the threshold as the respective eye movement's likelihood intersection. The GMM is may estimated by Expectation Maximization (EM) or online, using Bayesian inference (see Section 7.2) [12], [174], [175]

Hidden Markov Model Identification (I-HMM): The I-HMM implements an Hidden Markov Model (HMM) to decode the most likely sequence of fixation and saccades (extendable to smooth pursuits). Therefore, the gaze signal is interpreted as the output of a finite state machine, whereas the transition between the fixation and saccade state is modeled as a conditional likelihood. However, similar to I-BDT, the I-HMM's output is a likelihood of each sample to be part of fixation, saccade, and, if considered, smooth pursuit [172], [176], [177].

Principal Component Analysis Identification (I-PCA): The I-PCA utilizes a Principal Component Analysis (PCA) to determine the maximal dispersion axis in a time window. Like the I-DT, fixed thresholds separate the dispersion's major axis into fixations, saccades, and optionally smooth pursuits [178].

Area of Interest Identification (I-AOI): In contrast to velocity and dispersion thresholds, the I-AOI works on spatial scene boundaries. Consecutive eye-tracking samples falling into the same Area of Interest (AOI) are considered as a fixation sequence. A saccade is reported whenever a sample falls into an AOI that differs from the previous sample. I-AOI is frequently used for wordbook-based scanpath analysis, as elaborated in Sections 2.5.2, 8.1, and 8.2 [172].

Rayleigh Test Identification (I-VMPRay): I-VMPRay is a specialized filter for high-speed eye-tracking systems. Based on the Rayleigh test, I-VMPRay analyzes four spatial features in a sliding time window: dispersion, direction, displacement, and spatial range [179].

2.5.1.2 Pupillometry

The pupil's response in terms of change in diameter is a well-established measurement for cognitive load, particularly in psychology research. The pupil dilates in arousal and mental effort, yet also to brightness (light response) and near fixation (near response) [180].

Most eye-tracking systems report the pupil diameter as standard output as it is often a side product of the pupil center estimation. E.g., the previously introduced EyeRec(Too) – Section 2.2.3.5 – reports the major axis length of the fitted pupil outline ellipse as the pupil diameter in pixels. However, the major axis as pupil diameter is a gross simplification.

Although the main axis correlates to a spherical projected circle's radius (pupil on the eyeball), it disregards depth effects and assumes the eyeball to be perfectly aligned to the camera center. Hence, more sophisticated pupil diameter estimation uses a 3D representation of the eyeball and the detected pupil outline, e.g., Pupil Labs [53], or Tobii Pro [50]. However, estimating the user's eyeball is far from trivial, yet simultaneously the accuracy of 3D pupil diameter estimation crucial depends on the 3D eyeball model. Hence, accurate solutions require specialized hardware, i.e., glints, stereo vision, or structured light.

On the other hand, analyzing the pupil dilation often focuses on a normalized pupil diameter change over time, as discussed in Section 7.1. Hence, accurate physiological solutions often not required. Yet, important is a reliable correlation of the differential pupil size signal and the change in the actual pupil size. In a Nutshell, for most user studies, it is enough to know that the pupil significant dilates or contracts.

However, to distinguish the pupil dilation's principle in real-world scenarios stays challenging. Particularly in driving scenarios, light conditions may change rapidly, e.g., driving from a shadowed into a sun lighted area as frequently occur in urban scenarios. Yet, even more, challenging in twilight and night rides, where the eye may switches from mesopic to photopic vision depending on the street illumination and the gazed area (both underlying different pupil size baselines). Figure 2.5.2 shows a respective variation of the pupil diameter recorded in a night drive around the region of Brisbane, Queensland, Australia.

Such side effects are challenging to manage and evaluate, thus often confine the practical application of pupillometry to laboratory setups. However, with the emerge of broad available Head-Mounted Displays (HMDs), suitable conditions for pupillometry may straight forward implemented. HMDs isolate the user from external light sources, whereas the displayed stimulus is under full control of the researcher. Section 7.1 utilizes this fact to evaluate the driver's cognitive load with respect to an attention guiding assistant system.

2.5.1.3 Eyelid closure

The eyelid closure indicates the maximum distance between the lower and upper eyelid [19], [182]. However, some approaches address the eyelid closure as segmentation challenge and thus report the uncovered eye area as eyelid closure indicator, e.g., the proposed cycle GAN segmentation in Section 3.2 [9], [183].

The user's long term eyelid closure development is an important indicator of fatigue [184], [185]. Moreover, the short term analysis enables sufficient detection of blink, which is known as a valuable indicator for cognitive load and the performed task [31], [186], [187]. However, similar to pupillometry, the eyelid closure is significantly subjected to lightning conditions and very user individual. Nevertheless, the next-generation driver monitoring system probably uses eyelid closure to assist drivers by detecting fatigue and glares [21]–[23].

2.5.2 Scanpath

Scanpath analysis is part of the standard repertoire of every eye-tracking researcher. Accordingly, there is already a multitude of different approaches available, ranging from attention

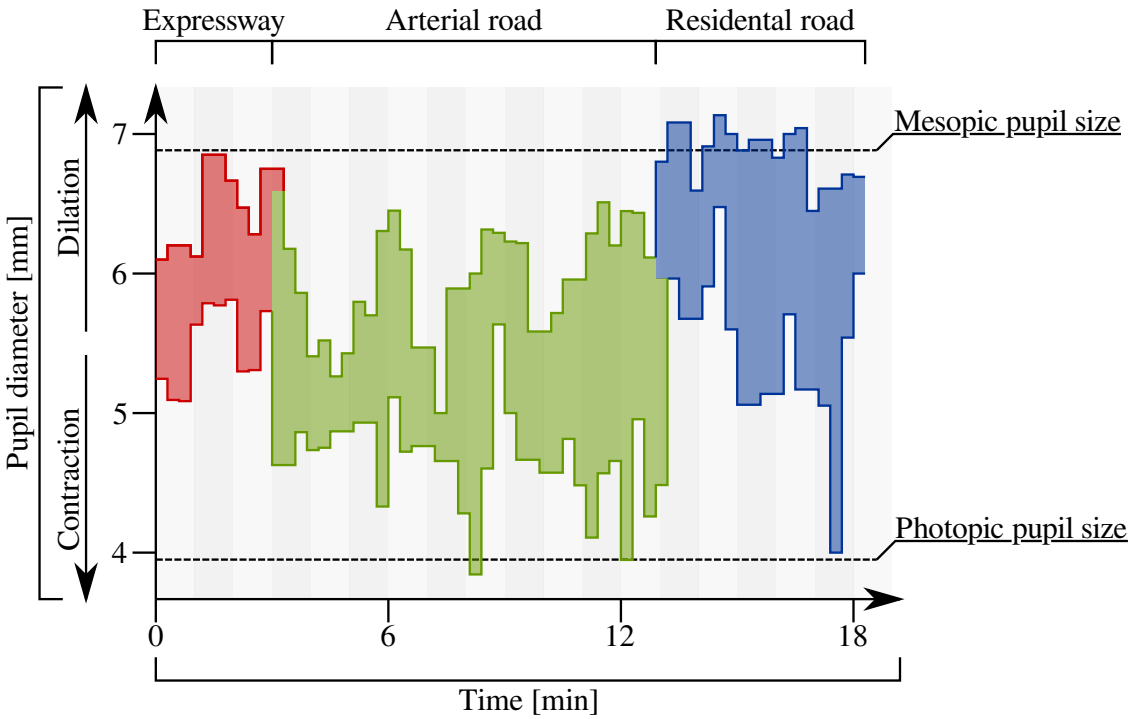


Fig. 2.5.2: The plot shows the variation of a driver's pupil diameter while a night drive on a expressway (red), an arterial road (green), and residental area (blue) in Brisbane, Queensland, Australia. The bins of 18 s each show a clear shift of the driver's pupil diameter over time, and traffic situation. The figure was resampled from A. Black et al. [181].

comparison to mixture models. The review from Anderson et al. provides a comprehensive overview of the state-of-the-art in 2014 [188]. Hence, the following focuses on the fundamental scanpath analyzing tools and complement Anderson et al.'s review with more recent approaches.

2.5.2.1 Scanpath Similarity

Besides the visual and manual evaluation of scanpaths [189]–[191], the most common automatic assessment is by their pair-wise similarity. One of the most well-known methods to calculate a scanpath's similarity is MultiMatch, introduced by Dewhurst et al. in 2012 [192]. MultiMatch calculates the scanpath distance in multiple dimensions, e.g., fixation location and duration. The strength of MultiMatch is that it can be applied directly to the scanpath (in the form of (x, y) coordinates). The similarity of two scanpaths is determined purely by their attributes. Moreover, MultiMatch may compensate for shifts in the eye-tracking signal, as well as reversed fixation orders, making it particularly robust in many application domains [192]–[194]. However, it also means that semantic information is not included in the scanpath distance.

Similarly, Eyenalysis maps the individual fixations of a scanpath to the closest of a second

scanpath. The similarity between the two scanpaths is then calculated by the spatial distance of the mapped fixations [195].

FuncSim, on the other hand, divides the scanpath into functional subunits and calculates the similarity of two scanpaths within the subunits [196]. The subdivision of the scanpath allows the input of semantic information.

2.5.2.2 String Similarity

Many approaches use a string representation to encode various scanpath properties and semantics into a one-dimensional signal (refer Section 8.1 and 8.2 for more details). Despite its weaknesses due to possible quantization errors, these approaches are still very successful in the calculation of scanpath similarity and subsequent classification of cognitive states, e.g., task recognition [67].

The mapping of the scanpath to a string is often done by Region of Interests (ROIs) in the stimulus. The ROIs can either be calculated automatically: e.g., using a static grid, fixation clusters, or with image processing methods [197]–[200]. They can also be labeled manually, which is generally a more accurate representation of the image or task semantics.

The gold standard in the calculation of string similarities is the Needleman-Wunsch algorithm, which calculates the costs of gap operations in both strings to align one string to another [201]. The Levenstein distance can be considered a special case of the Needleman-Wunsch algorithm. In contrast to the Needleman-Wunsch, only operations of the length one are allowed, which reduces the runtime complexity from $O\left(n^3\right)$ to $O\left(n^2\right)$ [202]. These approaches are highly investigated for scanpath comparison, and work very well on short scanpaths [203]–[207].

However, the string editing distance has some drawbacks: 1st) the runtime is not acceptable for long scanpaths. 2nd) the sequence of symbols in the string usually represents the fixation sequence. It is known that especially patterns in the transition between fixations contains valuable information about the executed task or expertise [203]. However, such patterns are not reflected in a mere string distance.

2.5.2.3 Subsequence matching

SubsMatch takes this into account and assesses the similarity of two scanpaths by the frequency of subsequences. Therefore, the string is subdivided into subsequences (words) using a sliding window. Each extracted word is counted in a dictionary. The scanpaths' similarity is then calculated as the distance over the words' [197], [208]. These words are particularly significant with regard to various classification problems, such as performed task – see Section 8.1 – or expertise [11], [67], [203], [209], [210].

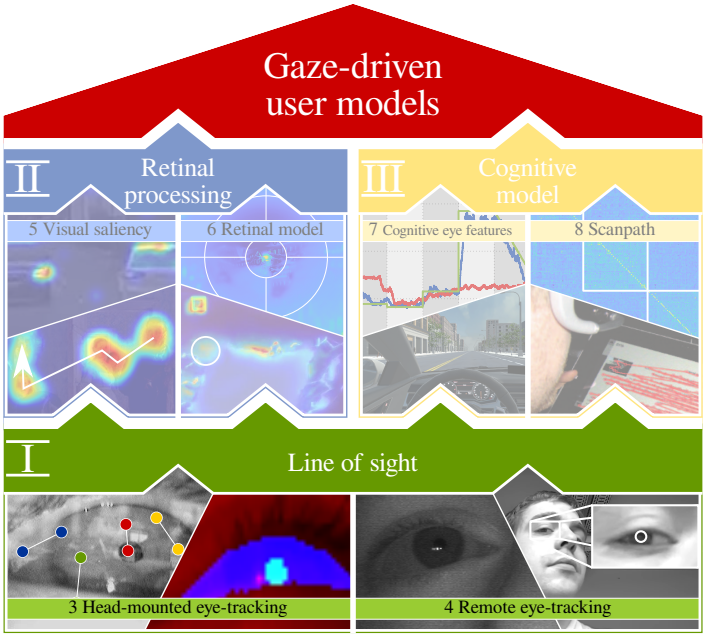
However, depending on the extracted word length and variety of symbols (alphabet) in the string, the dictionary may become extremely large. Thus, iterating over a union of two dictionaries to calculate the frequency differences can be very time-intensive. Conversely, researchers are restricted in the number of used ROIs, or the examined word length, to finish their work on time.

To reveal the full power of subsequence matching, the approach presented in Section 8.2 tackles the runtime challenge using MinHash. MinHash allows to calculate the scanpaths' pair-wise similarity as a Jaccard Index approximation of the string's subsequences, yet at a fraction of runtime on long scanpaths compared to SubsMath or Needleman-Wunsch [1].

However, reliable scanpath analysis requires robust and accurate pupil detection and gaze mapping. The next Part I introduces novel approaches for pupil detection in head-mounted eye-tracking in Chapter 3 and remote eye-tracking in Chapter 4.

PART I

Line of sight



The line of sight describes the imaginary line running from the foveal center of the eye through the pupil to the focused scene point. This line describes the center of spatial visual perception and corresponds to the scene area with the highest visual acuity – the fovea. This foveal perception accounts for the majority of the visually perceived stimuli and hence the most detailed yet spatially limited visual scene perception. In order to compensate for this, the human eye follows a temporal paradigm of a repetitive sequence of saccade and fixation. The saccade describes a rapid shift of the line of sight due to a rotation-like movement of the eyeball. During fixation, the line of sight remains aligned to a point in the scenery. During fixation, slower eye movements – so-called smooth pursuits – compensate for the inherent and object dynamics in the scene to track the point of regard, while the visual information is extracted and processed.

The selection of the fixated scene areas and the accompanying foveal visual scene perception is a sophisticated and highly developed cognitive process. Driven by action and intention,

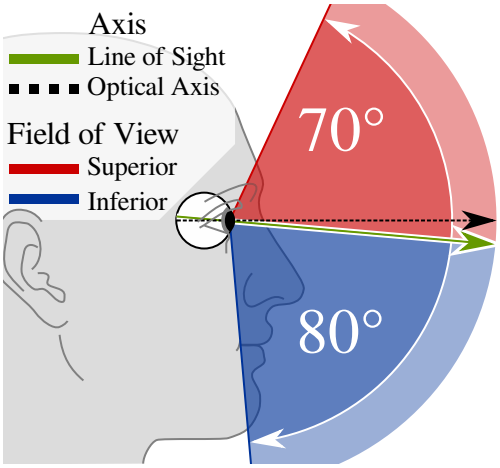


Fig. 2.5.3: The line of sight defines the center of the visual field of view. The the visual field extends from 60° nasal to 107° temporal (not depicted in the figure) and from 70° superior to 80° inferior around the line of sight. The optical axis is the line through the pupil center and the eyeball center. The deviation of the optical axis to the line of sight is individual, but usually not exceed 5°.

the visual exploration follows a strategy of optimizing the visual information extraction in terms of the eidetic environmental model, temporal relevance and significance of the respective scene area. Consequently, the spatio-temporal shift of the line of sight encodes information about cognitive processes and the eidetic world model and reveals insights into intention, action, cognitive load, and many more.

However, human visual perception is far from being limited to a straight line of sight. Apart from foveal vision, a large field of view is continuously monitored against the eidetic world model and assessed for possible fixation targets. This strongly parallelized and predominantly subconscious process is decisive for the temporal and spatial division of foveal attention. For instance, while driving, visual attention is mainly bound to the vanishing point of the own trajectory as it is, in most cases, the critical scene area in terms of the driving task. Simultaneously, other traffic relevant objects are tracked over the entire field of view and continuously compared and incorporated into the anticipated traffic situation – even without fixating them. However, the visual field extends from 60° nasal to 107° temporal and from 70° superior to 80° inferior around the line of sight (see Figure 2.5.3).

However, in order to evaluate the field of view in terms of the scene's visual perceivability beyond mere foveal perception, it becomes crucial to track the user's line of sight robustly and accurately. This part contributes four specific approaches to progress the tracking of the user's line of sight using eye-tracking technology. In the context of head-mounted eye-tracking, Section 3.1 deals with a detailed elaboration of pupil center detection using circular binary features and random ferns, followed by a cycle GAN approach for pupil and eyelid segmentation in Section 3.2. Regarding remote eye-tracking, Section 4.1 presents a sophisticated algorithm for 3D glint detection in a stereo camera setup, followed by an extensive evaluation of state-of-the-art pupil detection algorithms in Section 2.3.3.

CHAPTER 3

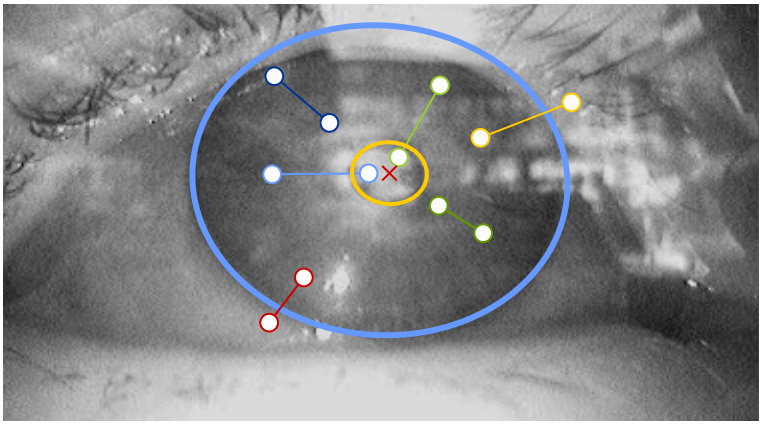
Head-mounted eye-tracking

Head-mounted eye-tracking enables egocentric tracking of the user's eye movements. The head-mounted frame allows close-up recordings of the user's eyes, with a consistent perspective, and independently of head movements. Hence, state-of-the-art head-mounted eye-tracking systems deal with high-resolution eye videos, yet without reference to the 3D scene system. Moreover, head-mounted eye-tracking systems usually provide a field camera, on which a detected pupil position is mapped as the user's POR. The user's POR is then extrapolated into the 3D scene camera system, e.g., by using additional sensors for head-tracking, or scene camera localization, as extensively elaborated in Section 2.2. However, this requires a robust and accurate detection of eye features, i.e., the pupil center. Yet, in head-mounted eye-tracking in the wild, state-of-the-art methods suffer particularly from external light sources, i.e., sunlight. These may induce challenging reflections, which manifest very diversely in the eye image. However, this multitude of interference affects decision-based algorithms, as introduced in Section 2.2, to their limits.

Hence, Fuhl & Geisler et al. introduces CBF, an intensity robust feature for machine learning-based pupil center detection [10]. Moreover, Fuhl & Geisler et al. also presented a novel cycle GAN framework for robust eye image segmentation and pupil detection, yet also for eye image refinement [9]. Applied as preprocessing step, The refinement cycle GAN pushes state-of-the-art decision-based algorithms' pupil detection rate to a competitive level with most recent machine-learning approaches.

The CBF and cycle GAN approach is presented in the following Sections 3.1 and 3.2, respectively.

3.1 Circular binary features for robust and real-time pupil center detection



ACCURATE and robust detection of the pupil center in the eye image is a crucial, but non-trivial computer vision task in head-mounted eye-tracking. However, in everyday scenarios, this includes various challenges such as reflections, varying illumination, distracting makeup, contact lenses, making it even challenging for humans to locate the pupil center.

Although PURE already achieves a remarkable $\sim$80 % detection rate, at an accuracy of $\leq$ 7 px, on a large convolute of datasets, it also implies that there is still room for improvement. Yet, manually, decision-based, created routines can only intercept to a certain extent, the diversity of potential error impacts and their manifestation. Table 3.1 shows some examples where even state-of-the-art decision-based approaches such as ElSe and PURE fail. At the same time, it shows that the machine-learning approaches such as PupilNet and the following presented CBF provide adequate performance, even in such particular challenging cases.

However, in practice, the major challenge for machine learning approaches are tough runtime requirements. Even a cost-efficient head-mounted eye-tracker, e.g., PupilLabs 2, provides frame rates up to 200 Hz. Thus, real-time pupil center detection has to require less than $\leq$ 8.3 ms. Lightweight, decision-based pupil detection algorithms, e.g., ExCuSe, achieves runtimes of $\sim$2.0 ms (500 Hz). Though even complex approaches, e.g., PURE levels at $\sim$5.5 ms (180 Hz). Such high runtime requirements require lightweight yet efficient machine learning approaches.

The following sections are supported by a prior publication from Fuhl & Geisler et al. at the conference on Eye-tracking Research & Applications (ETRA) in 2019 [10]. Section 3.1.1 introduces CBF, a random fern based detector using circular binary features. Section 3.1.2

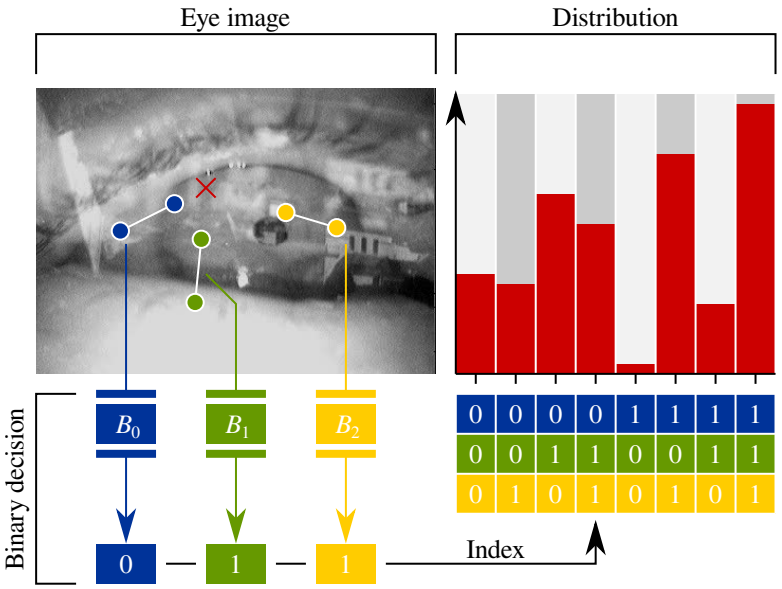


Fig. 3.1.1: Example of a pupil center probability lookup of a pixel (red cross). The example uses $N = 3$ comparators $B_0, \ldots, B_1$, which evaluate the lookup index to $\texttt{011} \longrightarrow 3$. The computed index enables a direct access to the conditional probability $P_j\left(\ldots \middle| B_0, \ldots, B_1\right)$ of a fern j. $P_j\left(\ldots \middle| B_0, \ldots, B_1\right)$ indicates the probability of the considered pixel to constitute a pupil center, on the condition that the comparators $B_0, \ldots, B_1$ evaluated to $\texttt{011}$.

provides an extensive evaluation and benchmark of the pupil center detection to the state-of-the-art. The final Section 3.1.3 discusses the limitations of this approach, and 3.1.4 concludes this section.

3.1.1 Method

CBF uses random ferns to break down a set of binary features to a unique index [211], [212]. The index is subsequently used for a lookup in a learned probability distribution of the training data. A single fern comprises multiple binary decisions that evaluate to either logical True or False. Whereby each logical evaluation directly infers the value of a respective bit in the index integer. Figure 3.1.1 illustrates the principle of a fern with $N = 3$ bits. B_0, B_1, and B_2 represent binary decisions, whose evaluation results in the bit sequences $\texttt{011}$, respectively, the index 3. In the case of CBF, $B_0, \ldots, B_{N-1}$ are simple *larger-* and *lower-than* pixel intensity comparators (where N is the number of comparators respective index bits). This makes this approach particularly robust to illumination changes.

The looked-up probability represents the conditional likelihood of a pixel to be the pupil center under the conditions of $B_0, \ldots, B_2$. Thus, a single fern conjugate several weak classifiers $(B_0, \ldots, B_2)$ to form one strong classifier. Furthermore, several ferns are combined by multiplication to form an even stronger classifier. Moreover, Villamizar et al. propose to use two conditional distributions [213]. One for the positive and one for the

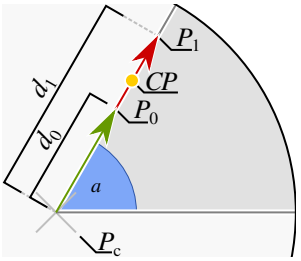


Fig. 3.1.2: P_c is the evaluated pixel. a indicates the randomly chosen angle. the feature points P_0 and P_1 are constructed, along the angle of a, and the two randomly generated distances d_0 and d_1. CP is the center point between P_0 and P_1, which serves as pivot to scale the characteristic as given in equation 3.2.

negative case, respectively, the considered pixel is a pupil center and not. Overall, this leads to the following likelihood definition:

$$\mathcal{L}(i) = \prod_{j=0}^{F-1} \frac{P\left(\text{Positive} \middle| B_0^{(j)}(i), \ldots, B_{N-1}^{(j)}(i)\right)}{P\left(\text{Negative} \middle| B_0^{(j)}(i), \ldots, B_{N-1}^{(j)}(i)\right)}, \tag{3.1}$$

where F is the number of ferns, $P(\ldots)$ is the probability that the pixel i is a pupil center, respectively, not a pupil center, given the condition of the calculated index $B_0^{(j)}(i), \ldots, B_{N-1}^{(j)}(i)$. Note: Fuhl and Geisler et al. used in their actual implementation a sum instead of the product, to stabilize the expression numerically. Furthermore, a small constant ϵ in the denominator prevents a division by zero.

Features The camera perspective of the eye and pupil may vary significantly according to the eye camera pose, optic, and resolution. In binocular eye-tracking, it is even familiar for the second eye's camera to be rotated around 180° (due to construction reasons). This requires a scalable and transformable set of features, to transfer a learned classifier to other setups. Thus, CBF constructs the features circular around an image position instead of using random image locations. Hence, first, a random angle a is defined, followed by two random distances d_0 and d_1 along the angular line of a. As shown in Figure 3.1.2, this feature construction leads to the key points P_0 and P_1. The key points are later used for the intensity comparison with comparator B. In the simplest case, P_0 corresponds to a point inside and P_1 outside the pupil. The feature construction is motivated by the elliptical shape of the depicted pupil. Whereas, the comparators correspond to edge detection.

However, to enable scale-invariance, the center point between P_0 and P_1 (CP), together with the angle a is selected as a reference point. P_0 and P_1 can be reconstructed from CP and a for any scaling s as follows:

$$P_0 = CP - \begin{pmatrix} s_x \cdot \cos(a) \\ s_y \cdot \sin(a) \end{pmatrix}, \text{ and } P_1 = CP + \begin{pmatrix} s_x \cdot \cos(a) \\ s_y \cdot \sin(a) \end{pmatrix}. \tag{3.2}$$

Whereby the feature transformation t is achieved by a mere shift of the center point CP: $CP_t = CP + t$

Training A challenge regarding training a conditional distribution is the strong impact of sample bias in the dataset. In the case of eye-tracking data, this problem is particularly

prominent, as the pupil remains in the same position over multiple samples during a fixation. As a workaround, only a tiny subset of the available training data was randomly selected to train a fern ($\sim$2000 images).

The training comprised several datasets and includes different eye image resolutions. Accordingly, the feature scaling s is set in proportion to the respective dataset resolution. Moreover, the eye images are previously downscaled by a factor of 2, to reduce computing time and suppress image noise. Positive features are collected around the labeled pupil center P_c, as shown in Figure 3.1.2. Moreover, positive instances were also sampled in a region of 7×7 pixels around P_c but weighted by the euclidian distance to P_c. Yet, negative instances are collected by randomly shifting P_c in the whole eye image.

The training phase is divided into two steps. The first step involves the training of a fern and a subsequent feature selection. The second step comprises the combination of multiple ferns into a strong classifier and refinement of the distributions using gradient descent.

Step (1) After randomly selecting a subset of training images, on each training image M previously randomly generated features were evaluated. Then, each feature is scored by a ranking metric as follows:

$$\text{Rank} = \left|(\text{True}_{\text{valid}} + \text{False}_{\text{Invalid}}) - (\text{False}_{\text{valid}} + \text{True}_{\text{Invalid}})\right|, \tag{3.3}$$

where $\text{True}_{\text{valid}}$ and $\text{False}_{\text{Invalid}}$ refer to the binary decision evaluated to True at a pupil center and to False at no pupil center. Accordingly, $\text{False}_{\text{valid}}$ and $\text{True}_{\text{Invalid}}$ correspond to negative cases, when the feature evaluates to False at a pupil center position, respective evaluates to True at no pupil center. In a nutshell, the rank counts the absolute difference between correct and incorrect classifications. The N highest-ranked features are selected as the fern feature pool. Yet, choosing N has a massive impact on the memory consumption of each fern. The evaluation in Section 3.1.2 bases on a fern size $N = 20$. Leading to a memory consumption of $4 \cdot 2 \cdot 2^N \approx 8.4\,\text{MB}$ for each fern.

The first step is repeated arbitrarily often, whereby each generated fern is evaluated by a validation set and stored in a fern pool. However, the validation set compromises positive samples in a region of 7×7 pixels, and negative samples in a region around 31×31 pixels around the labeled pupil center.

Step (2) The second step combines multiple ferns and fine-tunes their probability distribution using batch-based gradient descent. Therefore, random fern tuples are extracted from the fern pool and evaluated by a batch of 100 test images. For each missed pupil center, the positive distribution was increased by a learning rate of 10^{-2} – 10^{-4} (depending on the epoch). Accordingly, the positive distribution was reduced to the same extent if a pupil center was classified outside the 7×7 region of the labeled pupil center. Analogously, the negative distribution was trained inversely, but with a doubled learning rate.

After each batch, the modified ferns were tested on the validation dataset. If the performance increased, the detector was added to the fern pool. Thus, the fern pool is continually growing by tuned ferns until convergence. Finally, the F best-performing ferns are conjugated to a strong classifier, as shown in Equation 3.1.

Detector In a first step, the detector calculates the probability for a pupil center according to Equation 3.1 over the entire eye image. However, the image is heavily downscaled to improve runtime. The maximum of the resulting probability map is assumed as the coarse position of the pupil center.

In the second step, a patch of 31×31 pixels is cropped around the detected coarse pupil center. Within this patch, the pupil center is then fine-tuned by recalculating the probability map over each pixel. The pixel with the highest probability is reported as the detected pupil center.

The detection of the coarse to fine pupil center can also be divided into several scaling levels, with each level using separately trained ferns. However, the implementation by Fuhl and Geisler et al. is confined to two stages, both with separately trained ferns ($F = 2$), due to running time and memory consumption.

3.1.2 Evaluation

The CBF based random ferns were trained and evaluated over 4 distinct extensive datasets: ExCuSe ($38,000$ images) [62], ElSe ($55,712$ images) [63], PupilNet ($41,217$ images) [64], and labeled pupils in the wild (also referred as LPW, $130,856$ images) [66]. Whereas the datasets ExCuSe, ElSe, and PupilNet have been merged into one ($104,929$ images). Figure 2.2.3 shows sample images of the individual datasets along with a ranking of the respective challenge level.

Figure 3.1.3 shows the evaluation as a cumulative detection rate, along the state-of-the-art decision-based approaches PURE, ElSe, ExCuSe, and Świrski, and the deep learning CNN approach PupilNet. CBF significantly outperforms the state-of-the-art pupil recognition algorithms by more than 10 % in both evaluated datasets. Yet, as noted in Section 3.1.1, CBF still offers room for improvement. The evaluated configuration of $F = 2$ ferns, each with $N = 20$ comparators, corresponds to a minimal setup, limited by the used hardware for training and evaluation. However, with increasing hardware capacities, CBF could be parametrized even more generous.

Table 3.1 shows particularly difficult instances from the datasets ExCuSe and LPW. It is quite comprehensible that decision-based approaches fail completely in the illustrated situations. The respective instance in the ExCuSe dataset (II) provides a shallow contrast in the eye region. The other three instances in the LPW dataset have either extreme reflections (LPW3 file 1), high axial camera offset (LPW4 file 2), or disturbing objects between the camera and the eye. In this instance, the border of an eyeglass lens covers most of the pupil (LPW5 file 3). Yet, it turns out that in some cases, CBF is still able to track the pupil reliably (ExCuSe II and LPW3 file 1). However, even in cases where CBF also achieves only low performance, the detection rate still superior the state-of-the-art decision-based approaches.

As previously mentioned, the runtime of CBF is significantly related to its parameterization. Table 3.2 compares the duration of each evaluated pupil recognition algorithm. For evaluation, CBF was configured to operate online with 120 Hz eye-trackers (such as the PupilLabs 2 eye-tracker). Yet, CBF's actual runtime at $F = 2$ ferns and $N = 20$ comparators averages at 6.8 ms ($\sim$147.1 Hz). This is close to the performance of ElSe and PURE and even outperforms the CNN based PupilNet.

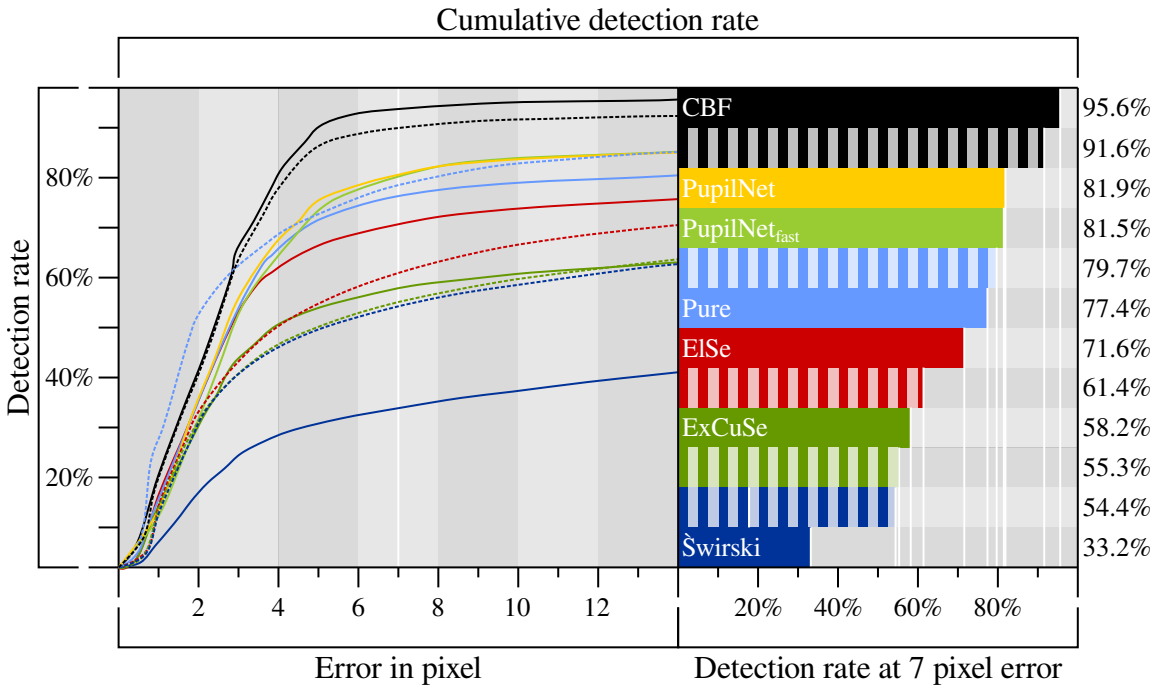


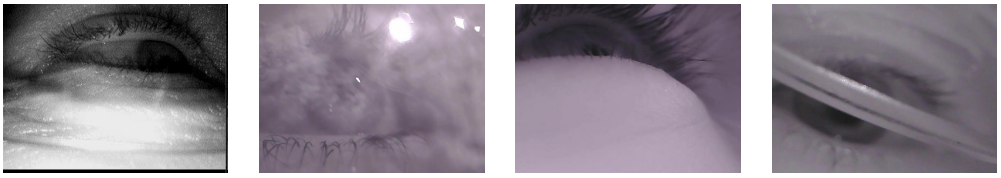
Fig. 3.1.3: The evaluated detection rate of the pupil detection algorithms form Świrksi et al., ExCuSe, ElSe, PURE, PupilNet, and the proposed CBF based random fern classification. The left diagram shows the conditional detection rate (vertical) for a certain accepted deviation in pixels (horizontal). The bar charts on the right side, compare the detection rates, at a tolerated deviation of < 7 pixels. The full colored series are evaluated on the joined datasets of ExCuSe, ElSe, and PupilNet ($104,929$ images). The dashed series were evaluated on the LPW dataset ($130,856$ images). The presented CBF approach significantly outperforms the state-of-the-art on both data sets.

3.1.3 Limitations

CBF, in conjunction with the machine learning approach of random ferns, demonstrates an impressive performance in its core task: detecting the pupil center. However, CBFs do not provide information about the actual structure of the pupil. In contrast to all considered decision-based approaches, which report the exact pupil contour as an ellipse. This does not limit the classical polynomial mapping of the detected pupil center to a scene gaze point. Even the back projection in 3D can be performed as elaborated in Section 2.2.4. Yet, the pupil center alone excludes the use of GRIP (see Section 2.2.3). Thus, the eye-tracking stack becomes vulnerable regarding slippages of the head-mounted eye-tracker.

Another significant limitation is the high memory and runtime requirements. Although a small configuration can successfully limit this. Still, to evaluate the full potential of CBFs, considerably more ferns with a large number of comparators would have to be used. Furthermore, larger datasets may offer a wider variety, which in turn increases the pupil

(a) ExCuSe II (b) LPW3 file 1 (c) LPW4 file 2 (d) LPW5 file 3



Dataset	ElSe	ExCuSe	PupilNet	PURE	CBF
(a)	65 %	40 %	80 %	29 %	99 %
(b)	8 %	1 %	–	30 %	92 %
(c)	7 %	3 %	–	9 %	52 %
(d)	1 %	2 %	–	1 %	20 %

Tab. 3.1: Instances of ExCuSe and LPW, where decision-based approaches fail. The upper row shows some examples of the especially challenging instances. The underneath table provides the detection rates of the evaluated pupil detection algorithms (PupilNet was not evaluated on the LPW dataset). Instance (a) and (b) demonstrate the robustness of CBF to different lighting challenges. However, even if the detection rate in instances (c) and (d) is considerably low, CBF's detection rate is still significantly higher than its competitors.

Tab. 3.2: The average processing time of a single pupil detection. All algorithms were evaluated on a input image resolution of 384×288 (ExCuSe, ElSe, PupilNet). The CBF approaches ranges between the decision-based approach ElSe and the deep leaning CNN approach PupilNet. CBF was deliberately parameterized to achieve a runtime of < 8.3 ms (corresponds to a frame rate of > 120 Hz).

Method	Runtime
CBF	6.8 ms
ExCuSe	2.0 ms
ElSe	6.5 ms
PURE	5.5 ms
PupilNet	7.0 ms
Świrski	4.0 ms

detection rate, especially in challenging eye images.

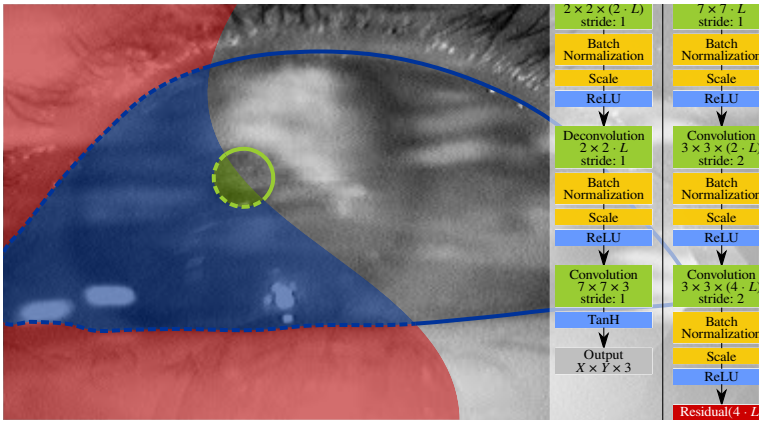
3.1.4 Conclusion

CBF was tested on a convolute of extensive datasets and showed an impressive accuracy above 90 %. Thus, CBF significantly outperforms all state-of-the-art methods. The used configuration enables an online operation mode (of the detector), at < 7 ms processing time and < 50 MB memory consumption of the ferns distribution tables. Yet, due to several pre-computed indices for runtime speed-up, the actual memory consumption is about < 3 GB.

The number of different positive samples in eye-tracking data is one of the most restrictive challenges. In the presented evaluation, training is therefore limited to only $2,000$ positive sample, to avoid over-fitting due to similar images during a fixation. However, the next

section tackles this challenge by introducing a cycle GAN architecture. The presented GAN is not only suitable for segmenting eye images, but also provides a data generator to generate artificial eye images, including reflections, noise, and other challenges. In this way, existing datasets may be further inflated, leading to a higher positive sample rate for the CBF training procedure.

3.2 Cycle GANs for pupil and eyelid segmentation



The previously introduced approach by Fuhl & Geisler et al., convincingly demonstrated the potential of machine learning approaches in computer vision-driven analysis of eye images. A lightweight detector based on random ferns and Circular Binary Features outperform all state-of-the-art decision-based pupil detection algorithms [10]. The same superiority is observed for the CNN based approaches PupilNet [64] and its sequel PupilNet v2.0 [65], DeepEye [81], and the real-time CNN from Eivazi et al. [83], although not throughout every dataset.

This evidence indicates that machine-learning-based approaches will supersede decision-based approaches for pupil detection in the foreseeable future. However, with the introduction of GAN in 2014 by Goodfellow et al. [214], and the cyclic loss function by Zhu et al. in 2017, the repertoire of machine-learning-based image processing grew by a powerful, versatile tool: the cycle GAN [215]. Cycle GANs transfer images across different domains, e.g., winter landscape to summer landscape, or eye image to segmentation, respectively, or augmented eye image to refined eye image.

This section introduces a cycle GAN framework to segment eye images to eyelids and pupils. The eye image segmentation enables the extraction of the pupil outline, center, size, eyelid closure, and eye corners. While the first two are vital features to predict the user's line of sight, the pupil size and eyelid are significant features of the user's cognitive state. Moreover, the introduced cycle GAN is also able to generate new artificial eye images based on the training dataset. Thus, already existing datasets can be enlarged. Finally, the same cycle GAN architecture is capable of refining eye images in order to improve the detection rate of state-of-the-art decision-based pupil detection algorithms. Thus, the cycle GAN removes challenging image augmentations, e.g., reflections or low image contrast, and improves the feature extraction of the subsequent pupil detection.

The following sections are supported by a prior publication from Fuhl & Geisler et al. at the International Conference on Computer Vision Workshops (ICCVW) in 2019 [9]. Section 3.2.1 introduces the cycle GAN architecture and its specific application to the various tasks. Section 3.2.2 provides an extensive evaluation and benchmark to the state-of-the-art, incorporating the eye image segmentation, pupil center detection, and image refinement. The final Sections 3.2.3 and 3.2.4 state the limitations and conclusion.

3.2.1 Method

A GAN architecture provides two competing networks, which work like a two-player minimax game [216]:

1st The generator G tries to create the most authentic representation of the output distribution B. Therefore, the generator learns to transform the input from the distribution A into the output distribution B, also known as style transfer: $G(a) \rightarrow B$, where $a \in A$.

2nd The discriminator D tries to find out whether its input is a generated image or a true image of the output distribution B: $D(b) \in B$, where $b \in B \cup G(A)$.

Consequently, the discriminator D is trained to maximize its accuracy in predicting whether a data point b comes from the data distribution B:

$$\mathbb{E}_{b \in B}\left[\log\left(D\left(b\right)\right)\right]. \tag{3.4}$$

In contrast to the generator G, which is trained to maximize the discriminator's prediction error:

$$\mathbb{E}_{a \in A}\left[\log\left(1 - D\left(G\left(a\right)\right)\right)\right]. \tag{3.5}$$

However, both networks are usually simultaneously trained, with their combined loss function given by the combination of equation 3.4 and 3.5 [214]:

$$\min_{G}\max_{D} V_{\text{GAN}}\left(D, G\right) = \mathbb{E}_{b \in B}\left[\log\left(D\left(b\right)\right)\right] + \mathbb{E}_{a \in A}\left[\log\left(1 - D\left(G\left(a\right)\right)\right)\right]. \tag{3.6}$$

Yet, this approach is hard to train, as the discriminator D tends to overfit, and thereby rejects everything. Therefore, Zhu et al. proposed in 2017 a cyclic loss function that takes both directions into account [215]. The transformation from A to B: $G_0(a) \rightarrow B$, as well as the transformation from B to A: $G_1(b) \rightarrow A$. However, this requires two generators G_0 and G_1, as well as to discriminator D_0 and D_1.

$$\begin{aligned}\min_{G_0,G_1}\max_{D_0,D_1} V_{\text{cycle GAN}}\left(G_0, D_0, G_1, D_1\right) = {} & V_{\text{GAN}}\left(D_0, G_0\right) \\ & + V_{\text{GAN}}\left(D_1, G_1\right) \ , \\ & + \lambda V_{\text{cycle}}\left(G_0, G_1\right)\end{aligned} \tag{3.7}$$

where $V_{\text{eye}}\left(G_0, G_1\right)$ is the cycle consistency loss. The objective of cycle consistency loss is to control that the two generators perform a consistent transformation in both directions. Thus,

in the optimal case, $G_1(G_0(b)) = b$ and $G_0(G_1(a)) = a$ evaluate to true. Accordingly, the cycle consistency loss is defined as follows:

$$\lambda V_{\text{cycle}}(G_0, G_1) = \mathbb{E}_{b \in B}\left[\|G_1(G_0(b)) - b\|\right] + \mathbb{E}_{a \in A}\left[\|G_0(G_1(a)) - a\|\right], \tag{3.8}$$

where λ controls the relative significance of the cycle consistency.

Network architecture Figure 3.2.1 shows the proposed architecture of the generators G_0 and G_1. The principal structure of the networks corresponds to the architecture proposed by Zhu et al. [215]. The feature extraction consists of three convolution blocks followed by batch normalization and a Rectifier Linear Unit (ReLU). Instead of pooling, the proposed network uses the stride parameter to downscale the input. The style transfer is implemented by N consecutive residual blocks, with constant block depth. Each residual block consists of three convolution layers, followed by a batch normalization layer and a ReLU layer. Finally, the new image is generated using transposed convolution layers.

The discriminators D_0 and D_1 comprise four consecutive convolution layers with batch normalization and ReLU layer. The last convolution layer reduces the input to a 1D output.

Model configuration The parameter L controls the block depth of convolution layers in the network. Whereas, the parameter N defines the amount of residual block in the central part of the generator. For all use cases, the model was configured with $N = 5$ residual blocks and a block depth of $L = 64$.

Besides the mere network parameters, the input size of the network is also essential. For the evaluation of runtime and memory consumption in Section 3.2.2, the net was tested on inputs with resolutions of 32×32, 64×64, 96×96, 128×128, and 256×256 (see also Table 3.6). However, for eyelid and pupil segmentation, the input was scaled down to a resolution of 64×64. Whereas to detect the pupil center, the input was scaled to a resolution of 128×128, to achieve a more accurate localization. The refinement GAN uses the highest resolution of 256×256, to reduce the impact of the upscaling operation to the original size of the image.

Eye image segmentation The eye image segmentation GAN is trained using paired images. Thus, for each training eye image, the respective segmented version was used as ground truth[1]. The input eye image of the network is a 3-channel grayscale image (each channel has the same value), with a resolution of 64×64 pixels. The output image is also a 3-channel image, where each channel representing a semantic segmentation:

- 1st channel: Provides the input image as a 1-channel grayscale (shown in red).
- 2nd channel: Indicates the pupil area (shown in green).
- 3rd channel: Defines the visible part of the eye (shown in blue).

[1] It is also possible to use unpaired images for training, yet the accuracy improves by $\sim$10 % when using paired image examples (see Table 3.3).

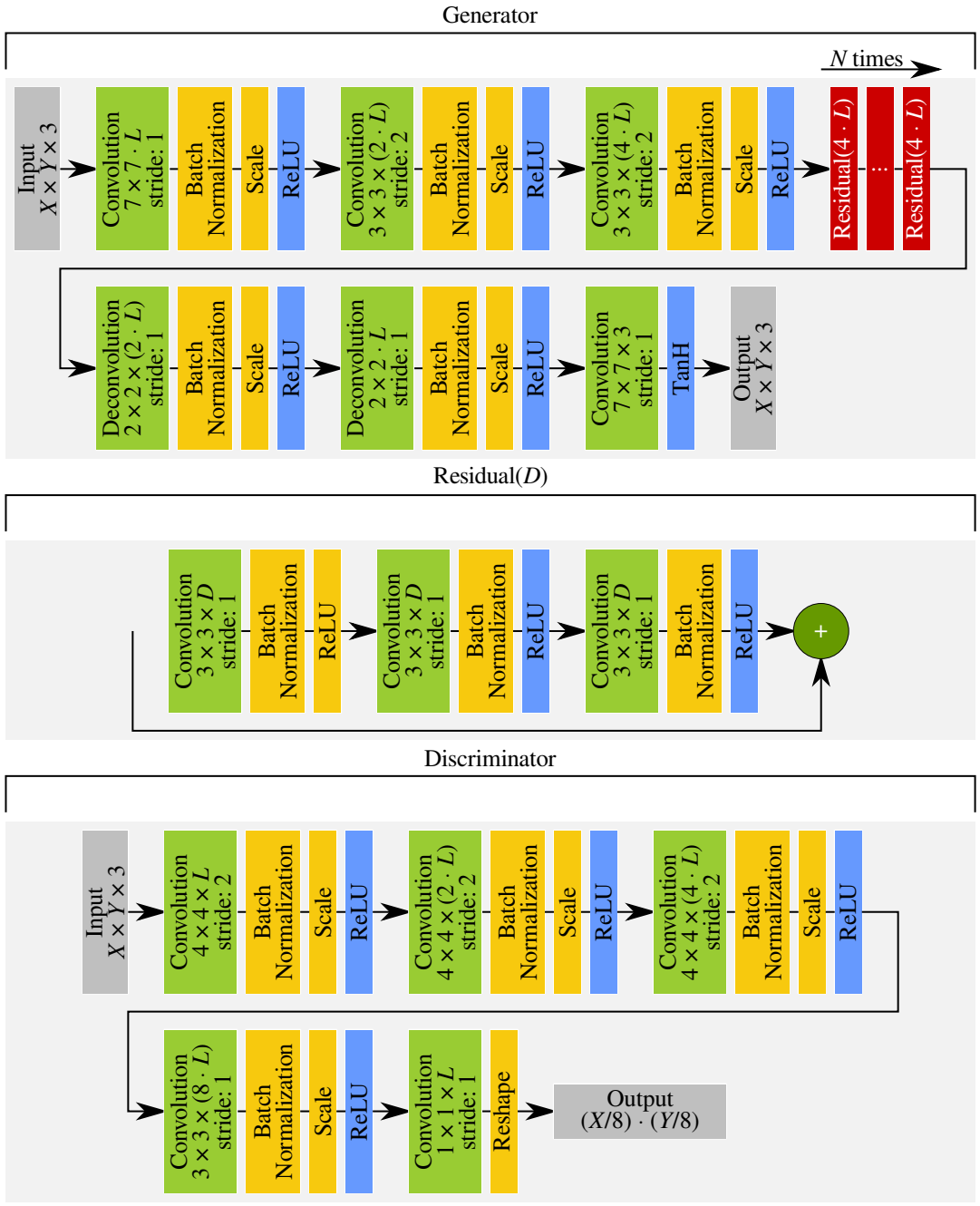


Fig. 3.2.1: Architecture of the used generator G_0, G_1, and discriminator D_0, D_1.

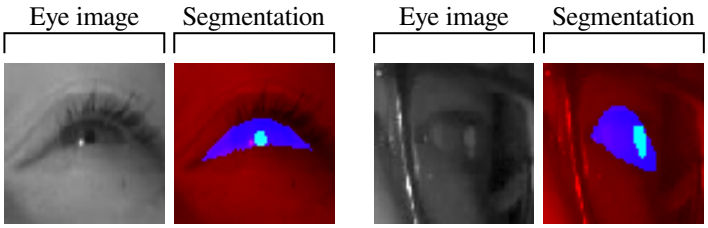


Fig. 3.2.2: Input and output eye image pairs, of the trained cycle GAN after ∼200 epochs.

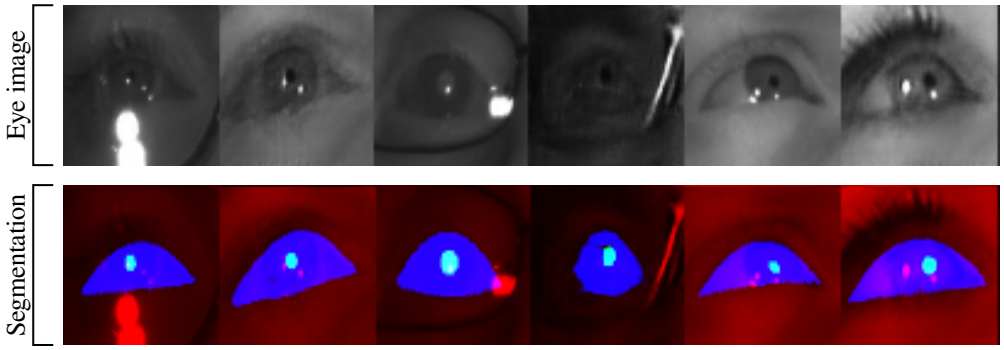


Fig. 3.2.3: Artificial eye images and their respective segmentation, as generated by the proposed cycle GAN architecture.

This configuration simplifies the evaluation of the segmentation based on the output and enables the network to reconstruct the input image similar to a deep autoencoder. Figure 3.2.2 shows the segmentation results of the cycle GAN after ∼200 epochs. The input image was wholly reconstructed and segmented.

Eye image generation The input dataset is divided into two distinct subsets to train the eye image generator: the input dataset, and the target domain dataset. Each image of the two datasets was constructed analogously to the output image of the segmentation GAN (3 channels: grayscale image, eye segmentation, and pupil segmentation). Thus, the input and output contained the image and the segmentation. Therefore, the GAN generates not only new eye images but also the corresponding segmentation. Thus, the ground truth labels for, e.g., pupil center, can be deduced from the newly generated eye images. The training is performed on the two datasets (unpaired images).

Figure 3.2.2 shows some examples of the generated images. Interestingly, the GAN learns to add a bright pupil (third picture in the upper row), eyelashes (last picture in the upper row), or glass frames with reflections.

Eye image refinement The eye image generator GAN may increase the quantity of training data for human-related challenges, like eyelashes, eye shape, pupil size, moles, and many more. Yet, technical challenges related to image acquisition, such as noise, reflections, and illumination, are not covered, unless they are sufficiently represented in the training data. Moreover, (recorded) eye images with particular challenges are usually hard to segment by hand.

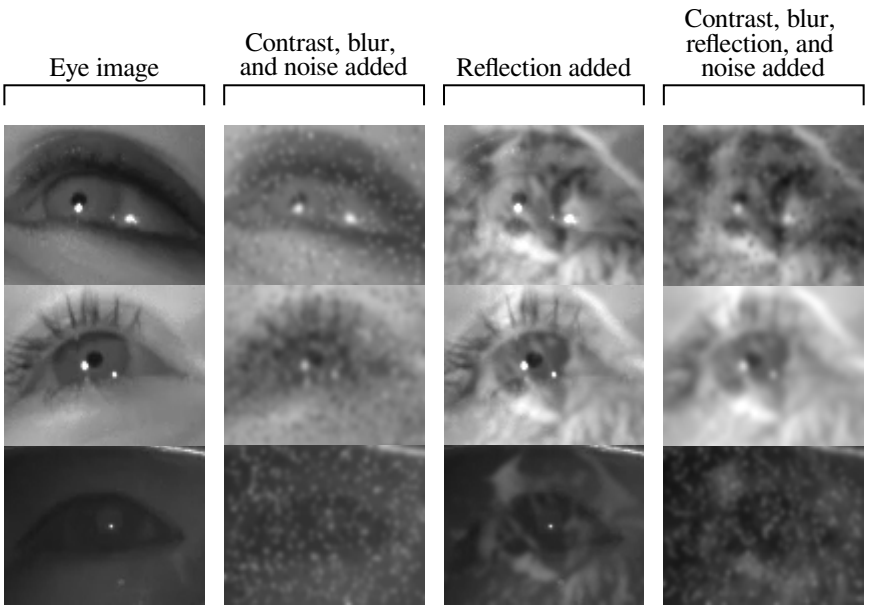


Fig. 3.2.4: Augmented images for refinement and augmentation training.

However, a generator G_0 for eye image refinement also offers its counterpart G_1: a generator for eye image augmentation. Therefore, the dataset was artificially expanded by random noise, image patch covering, blurring, contrast variations, and reflections. Figure 3.2.4 provides some examples of the artificial image augmentation. The first column contains the raw input images. The second column shows the applied modifications in terms of image contrast, blur and noise. In the third column, the eye images are augmented by a cat image reflection. The fourth column shows the combination of all augmentations: Reflections, noise, blur, and contrast changes. The reflections are added analogous to the approach of Wan et al. [217]. Therein, reflections are assumed to be a blurred additive of a second image. The training uses unpaired images from the raw data set as input, and from the artificially augmented images as the target domain.

Dataset The cycle GAN was trained on a dataset comprising $16,200$ hand-labeled images with a resolution of 1280×752 pixels from a total of 6 subjects [218]. Additional $\sim 25,000$ images from 9 subjects ($\sim 41,200$ images in total from 15 subjects), which are not publicly available, were used to evaluate pupil center detection (based on the segmentation GAN). Figure 3.2.5 shows images from the used dataset. Each image provides labels of the eyelids, and if visible, the pupil contour. The dataset comprises images with strong reflections, open and closed eyes, and high axial camera offsets.

3.2.2 Evaluation

The evaluation of the individual GANs was performed in a cross-subject paradigm. Therefore, each network was trained on the images of 14 of the 15 subjects. The images of the

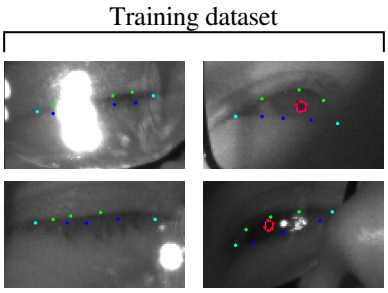


Fig. 3.2.5: Examples from the labelled training dataset. The data set consists of NIR recorded images, and includes various challenges, such as reflections (especially on glasses), closed eyes, and high axial camera offsets [218].

excluded subject formed the evaluation dataset. Thus, the data of each subject was used once for evaluation and 14 times to train 15 network instances.

The proposed GAN segmentation was benchmarked against three state-of-the-art deep learning approaches for pupil or eyelid detection:

1. The landmark detection provided by DLIB [219], [220]. Training and evaluation were performed using the same paradigm as the proposed cycle GAN.

2. The Fast and Robust Eyelid Recognition (FRED) by Fuhl et al., which provides an already pre-trained detector. The evaluation was directly performed against all 15 recorded subjects [221].

3. The pupil segmentation CNN DeepVOG [82]. Like FRED, DeepVOG provides a pre-trained detector; hence the evaluation was performed directly on the 15 subjects.

Moreover, the segmentation GAN was also evaluated against two decision-based, state-of-the-art pupil detection algorithms, namely ElSe and Boosted-oriented Edge Optimization for Robust, Real-time Remote Pupil Center Detection (BORE). Yet, the evaluation is limited to the pupil segments, as both reports only the pupil contour and no eyelid.

The evaluated performance is based on the Jaccard index as an accuracy metric:

$$\text{Accuracy} = \frac{\text{GT} \cap \text{DT}}{\text{GT} \cup \text{DT}}, \tag{3.9}$$

which describes the intersection between the detected area DT and the ground truth area GT, divided by their union. This is a common metric to evaluate segmentation algorithms. An accuracy of 50 % is considered a good result.

Table 3.3 shows the results of the state-of-the-art methods, along with the proposed segmentation GAN. The segmentation GAN was evaluated on the original data set, on the generated dataset, as well as on the joined data set (original cup generated). The proposed GAN architecture significantly outperforms state-of-the-art approaches. The best performance was achieved by using the original dataset together with the generated data, and a paired image training. In this configuration, the proposed approach outperforms the state-of-the-art with $> 30\,\%$ in the eyelid segmentation and $> 10\,\%$ in the pupil segmentation. However, it must be noted that the competitors' first four algorithms (DLIB's landmark detection, FRED, ElSe, and BORE) were designed to run in real-time on a single CPU. Yet, the segmentation GAN requires $\sim$18 ms on a high-end GPU (NVIDIA GTX 1050Ti) at almost 1 GB memory consumption.

Method		Eyelid	Pupil
DLIB		52 %	–
FRED		52 %	–
ElSe		–	33 %
BORE		–	65 %
DeepVOG		–	22 %
GAN	Original dataset, unpaired training	70 %	62 %
	Original dataset, paired training	79 %	72 %
	Generated dataset, unpaired training	71 %	63 %
	Generated dataset, paired training	78 %	72 %
	Original dataset ∪ generated dataset, paired training	84 %	78 %

Tab. 3.3: Average Jaccard Index per algorithm cross validated on the dataset from Fuhl et al. [218].

The segmentation GAN was also applied as a pupil center detector. However, an image resolution of 128×128 was used to increase accuracy. The pupil center was derived as the center of the segmented pupil (center of mass). As a benchmark, the decision-based approaches ElSe [63], ExCuSe [62], Świrski [60], and PURE [57] were used. Furthermore, the previously introduced CBF approach [10], together with the CNN-based approaches PupilNet [64], DeepEye [81], DeepVog [82], and the approach of Eivazi et al. [83], were used as deep learning competitors. Yet, in contrast to the above segmentation evaluation, the pupil center evaluation was performed on the data sets of ElSe [63], ExCuSe [62], PupilNet [64], Świrski [60], and LPW [66]. This leads to more comparability to Sections 3.1.2 and 2.2.2.

Table 3.4 shows the detection rate of all benchmarked algorithms. Whereas, a deviation of < 5 px was considered to be correctly detected[2]. The GAN approach shows comparable performance to the CBF approach and therefore outperforms other competitors analogous to CBF. However, the GAN approach requires significantly more runtime than CBF ($\sim$33 ms vs. $\sim$6.8 ms).

However, the cycle GAN was not designed to detect the pupil center. Rather, the GAN transfers images across different image domains. Thus, the GAN is able to pre-process images for a subsequent state-of-the-art pupil center detection. This image refinement step was trained on the artificially augmented images, as shown in Figure 3.2.4. Thus, one generator G_0 was trained to create new augmented eye images, whereas the second generator G_1 tried to remove the eye image augmentation. Table 3.5 shows that this refinement (using G_1) step, may significantly improve the pupil detection rate of decision-based approaches. As an extreme case, the pupil detection algorithm of Świrski et al. achieves a 39 % higher detection rate, and thus shows similar detection rates as the state-of-the-art algorithms ElSe and PURE.

Table 3.6 shows the runtime performance of the proposed GAN architecture at different

[2]Note, in Sections 3.1.2 and 2.2.2, a deviation of < 7 px was considered as acceptable.

Method		Datasets: ElSe, ExCuSe, and PupilNet,	Datasets: Świrski	Datasets: LPW
ElSe		67 %	81 %	54 %
ExCuSe		54 %	86 %	50 %
Świrski		30 %	77 %	49 %
PURE		72 %	78 %	73 %
CBF		91 %	–	87 %
PupilNet		76 %	–	–
DeepEye		83 %	54 %	50 %
Eivazi et al.		79 %	74 %	84 %
DeepVOG		43 %	52 %	62 %
GAN	Original dataset, paired training	87 %	89 %	85 %
	Generated dataset, paired training	91 %	93 %	89 %

Tab. 3.4: Average detection rate over all subjects, over the datasets provided with ElSe [63], ExCuSe [62], PupilNet [64], Świrski [60], and LPW [66]. The accepted deviation is < 5 px. In other words, a pupil center detected at < 5 px deviation (euclidean distance) to the labeled pupil center, was considered as correct[2].

Method	Datasets: ElSe, ExCuSe, and PupilNet,	Datasets: Świrski	Datasets: LPW
ElSe	74 % (+7 %)	81 % (+0 %)	73 % (+19 %)
Świrski	69 % (+39 %)	79 % (+2 %)	71 % (+22 %)
PURE	74 % (+2 %)	79 % (+1 %)	74 % (+1 %)

Tab. 3.5: Pupil center detection rate of state-of-the-art decision-based algorithms, using the eye image refinement of the proposed GAN as pre-processing. The detection rate was processed over all instances of the datasets, provided with ElSe [63], ExCuSe [62], PupilNet [64], Świrski [60], and LPW [66]. The classification was considered as valid, if the predicted pupil center deviated < 5 px from the label[2].

Input	Execution		Training	
Resolution	Runtime	Memory	Runtime	Memory
32×32	16.7 ms	0.80 GB	341 ms	1.50 GB
64×64	17.4 ms	0.85 GB	418 ms	1.70 GB
96×96	24.0 ms	0.90 GB	515 ms	2.00 GB
128×128	33.0 ms	1.00 GB	657 ms	2.30 GB
256×256	91.4 ms	1.50 GB	1715 ms	5.0 GB

Tab. 3.6: Average processing runtime (10^4 samples) on a NVIDIA GTX 1050ti GPU (4 GB memory), at different input and output resolutions. The batch size was to 1 (single image).

input sizes. The GAN architecture shown in Figure 3.2.1 was evaluated with the parameters $N = 5$ and $L = 64$. Obviously, the presented approach requires substantial hardware resources.

3.2.3 Limitations

The major limitation of the presented cycle GAN is undoubtedly the high requirement for hardware resources. First, the proposed approach requires a modern GPU, compared to the considered state-of-the-art methods, which run on GPUs. Second, the 4 GB memory of the NVIDIA GTX 1050ti, used in for the evaluation, is already insufficient to train the GAN thoroughly in GPU memory at an input resolution of 256×256. Thus, Caffe starts to swap memory between host and GPU, which significantly increases the runtime (see Table 3.6). Furthermore, the cycle GAN does not achieve a significant improvement in the detection of the pupil center compared to the CBF approach. However, the GAN requires more than three times higher runtime.

Nevertheless, the proposed GAN architecture shows promising results in the segmentation of eye images. In this smallest configuration (32×32 pixel input resolution), the GAN segments the images in $\sim$33 ms ($\sim$30 Hz). This is sufficient, e.g., to monitor the eyelid closure of a driver over a long period, to detect fatigue or other irregularities. It is also expected that increasingly powerful hardware will be available in the near future. Moreover, the data generation GAN enables researchers to inflate their datasets by new artificial eye images, and thus empower their machine-learning approaches. Vice versa, the image refinement pushes fast and lightweight decision-based algorithms to the performance level of the latest machine learning approaches. Even if the accuracy of, e.g., CBF is not achieved yet, this hybrid approach provides the advantage to run, e.g., PURE. In contrast to CBF, PURE detects the entire pupil contour. In turn, this enables GRIP, to process the gaze point, leading to more slippage robust eye-tracking stack.

3.2.4 Conclusion

This section introduces a true all-rounder. The presented cycle GAN architecture is able to segment eye images, detect the pupil center, generate new artificial eye images, and to remove augmentations on eye images. The extensive evaluation reveals that the GAN is on a par, or even better than the state-of-the-art.

The eye segmentation outperforms the state-of-the-art, and thus enables an even more precise evaluation of the eyelid closure or pupillometry, e.g., in the field of driver monitoring. Furthermore, the eye image generator creates authentic artificial eye images, along with their segmentation. For instance, limited datasets can be artificially extended, which enables a more precise evaluation of decision-based approaches, but also the training of complex machine learning approaches. The elimination of eye image augmentations, e.g., reflections, enables the application of decision-based approaches to challenging datasets, almost at the same performance level as the state-of-the-art approaches of machine learning. Altogether, cycle GANs represents a valuable multitool in the eye-tracking research domain.

However, this chapter focused on the extraction of eye features in head-mounted eye-tracking images. In contrast to remote eye-tracking images, head-mounted eye-tracking usually provide high and equal resolution, well-aligned perspectives of the user's eye. The next Chapter 4 addresses challenges of eye and pupil detection in remote eye-tracking.

CHAPTER 4

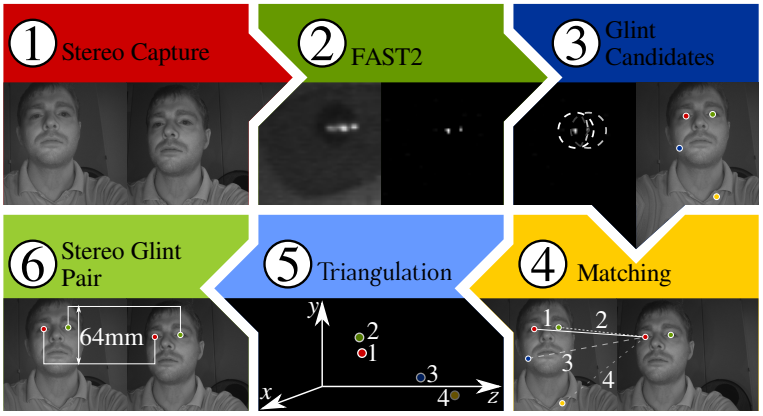
Remote eye-tracking

Remote eye-tracking is the art to track a user's eye position from an external viewpoint. In contrast to head-mounted eye-tracking, the remote eye-tracking system's cameras are stationary placed in the scene. Thus, the user's eye is recorded from a fixed perspective at some distance. However, this enables remote eye-tracking systems to evaluate the user's line of sight straightforward in a 3D reference coordinate system, yet implies may unfavorable perspectives, e.g., when the user performs exhaustive head movements. Moreover, even remote eye-tracking systems usually implement multiple cameras with higher resolutions (temporal and spatial) and higher pixel depth, the distance to the user leads to low-quality eye images in terms of pixel dynamic and resolution, compared to head-mounted eye-tracking systems. Furthermore, remote eye-tracking systems arise the challenge to detect the user's relevant features, i.e., eyes or facial landmarks, in a relatively large image. However, this requires particularly efficient methods to extract the essential features for the line of sight processing from an extensive search area, yet, fast, robust, and accurate.

A widespread technique is the use of glints (see also Section 2.3.2). Glints are specular reflections (1st Purkinje picture), caused by the eyeball's moisty surface. Therefore, artificial glints are induced by additional light sources in the scene (usually in IR). In laboratory conditions, the glints are straightforward detectable as an intensity peak. Thus glints used for fast and robust detection of the user's eye region. However, in the wild, glints may be caused by a multitude of light sources and may appear on all specular reflecting surfaces.

The following Section 4.1 introduces a novel probabilistic approach from Geisler et al. to detect and filter glints, even on challenging conditions. Moreover, Section 4.2 provides an extensive evaluation of state-of-the-art remote eye-tracking pupil detection algorithms for low-resolution eye images.

4.1 Realtime glint detection in remote eye-tracking



State-of-the-art glint-based approaches are sufficient for well-controlled (laboratory) settings and with a fixed number of static light sources. Under these constraints, it is possible to extract glints straightforward by investigating histograms [222] or applying edge detectors [93], [223]. Yet, these approaches fail beyond laboratory conditions, since strong light gradients, varying lighting, and reflections, as emerge especially in driving scenarios, may interfere with glint detection. This section addresses the aforementioned challenges and provides a method for robust yet fast glint detection for remote eye-tracking based on a probabilistic method, which provides a profound confidence measure enabling subsequent systems (e.g., prediction of a line of sight) to assess the reliability of their model input data. More importantly, as a by-product, the probabilistic model estimates the head pose in 5 degrees of freedom (3D position, yaw, roll).

The following sections are supported by a prior publication from Geisler et al. at the International Conference on Robotics and Automation (ICRA) in 2018 [5]. Section 4.1.1 provides a robust filter to extract glint candidates from stereo images inspired by FAST-like features. Subsequently, a stereo matching selects the most probable glint pair for each eye based on a sophisticated, flexible, and adaptive probabilistic model and bayesian inference. Section 4.1.2 introduces a new dataset designed as particularly challenging for glint detection. Furthermore, an extensive evaluation benchmarks the proposed glint filter and compares the proposed FAST-like glint extractor to several state-of-the-art approaches. The final Sections 4.1.3 and 4.1.4 address the limitations of this approach and conclude this section.

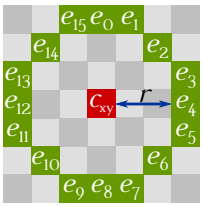
4.1.1 Method

The proposed glint detection method consists of two stages: 1st) glint candidates are extracted from a stereo input image by a spatial peak detection. This step is called FAST2 since it was

inspired by FAST feature detection. 2nd) after detecting potential glint candidates, a stereo matching and triangulation allows filtering the most likely glint pair in 3D by applying an adaptive probabilistic model. The second filter step allows the filters in the first stage to be adjusted less restrictively, which on the one hand, leads to a larger amount of extracted glint candidates due to reflections, other light sources, or other noise but also considers weakly pronounced glints. Thus, it is possible to detect glints, even in very challenging lighting conditions. In addition, the probabilistic model provides a suitable confidence measure, which enables the detection of a lack of a glint. The algorithmic pipeline comprises six successive steps, whereby the steps ① to ③ address the extraction of glint candidates, and steps ④ to ⑥ implement the stereo matching and the probabilistic model. The following elaborates each step in detail:

① Stereo capture The proposed probabilistic model operates on glints in a 3D coordinate system. Hence, the following steps presume a stereo camera setup, as illustrated in Figure 2.3.1, to retrieve depth information. The setup consists of two planar aligned cameras with a fixed vertical and horizontal offset. The user's head is facing the cameras; thus, both glints are visible in both camera images. Thus, the glint's intensity peaks indicate the relevant camera image area of the user's eyes, respectively, head.

② FAST2 The proposed filter highlights glints in an image by a center-surround difference. This is driven by the assumption that areas in the image corresponding to glints are significantly brighter than their closer neighborhood. A glint is defined as a group of related pixels that are at least brighter than a threshold t_{fast2} compared to all surrounding pixels in a certain radius r_{fast}. The difference of the maximal intensity in the surrounding pixels $e_0, \ldots, e_n$ to the considered pixel c_{xy} is normalized by the average intensity $\hat{c}_{xy}$ around c_{xy}, as shown in Equation 4.1. The resulting response k_{xy} is subsequently thresholded by t_{fast2}, leading to the sparse response map $m_{xy} \in M$. The threshold t_{fast2} controls the sensitivity of the glint detection (smaller t_{fast2} results in a more sensitive glint detection). t_{fast2} is regulated in an adaptive way, by decreasing it if less than a certain number of glints were detected or no two corresponding glints could be selected in the last iteration in step ⑥. Accordingly, t_{fast2} is increased if too many glints were found, and glint detection was successful. The radius r_{fast2} controls the spatiality of glint detection. It should be larger than the maximal expected glint radius, but as small as possible.



$$k_{xy} = \frac{c_{xy} - \max(e_0, \ldots, e_n)}{\hat{c}_{xy}}$$
$$m_{xy} = \begin{cases} k_{xy} & t_{\text{fast2}} \leq k_{xy} \\ 0 & t_{\text{fast2}} > k_{xy} \end{cases}, \tag{4.1}$$

where $c_{xy} \in I$ is the raw, and $\hat{c}_{xy} \in \hat{I}$ is the average filtered input image. $k_{xy} \in K$ contains the dense, and $m_{xy} \in M$ the sparse response map of the feature extraction.

③ **Glint candidates** After extracting the above features, the pixels are sorted by their response. The n highest responses are considered as glint candidates. Taking into account that glints may spread over more than one pixel (depending on r_{fast2}), new glint candidates are suppressed by weighting the distance to all already found candidates exponentially based on the following condition:

$$g_i = \begin{cases} m_i & \text{if } t_{\text{dist}} < \sum_{j<i}^{j=0} e^{-\frac{\|m_i - g_j\|^2}{2\sigma^2}} \text{ and } m_i \in R \\ \emptyset & \text{otherwise} \end{cases}, \tag{4.2}$$

where $g_i \in G$ is the i-th glint candidate and $m_i \in M$ the i-th element of the sorted pixels. t_{dist} and σ are two competing parameters to adjust the minimal allowed distance of two glints. Setting $\sigma = \sqrt{0.5} \cdot r$ and $t \leq e^{-1}$ ensures a minimal distance of r of two glint candidates. R is the region of interest where the head of the subject is expected. Without further knowledge, R is chosen to cover the whole input image.

④ **Matching** After extracting a set of glint candidates $g_{\text{left } l} \in G_{\text{left}}$ and $g_{\text{right } k} \in G_{\text{right}}$, in the left and right frame, respectively, the following step matches the two sets by their maximum likelihood $\mathcal{L}_{\text{match } l,k}$, taking into account the similarity of the patches around the glints, as well as geometric constraints derived from the camera setup. For this purpose, the following defines a series of likelihood functions that model the requirements as non-normalized exponential distributions.

The similarity likelihood $\mathcal{L}_{\text{BRIEF } l,k}$ of two glints is defined by the distance of the Binary Robust Independent Elementary Feature (BRIEF) descriptor for a patch around the glint candidate extracted from the left, respectively, right frame:

$$v_{\text{BRIEF } l,k} = \delta(f_{\text{BRIEF}}(g_{\text{left } l}), f_{\text{BRIEF}}(g_{\text{right } k})), \tag{4.3}$$

and

$$\mathcal{L}_{\text{BRIEF } l,k} = e^{-\frac{v^2_{\text{BRIEF } l,k}}{2 \cdot \sigma^2_{\text{BRIEF}}}}, \tag{4.4}$$

where $\delta(f_{\text{BRIEF}}(g_{\text{left } l}), f_{\text{BRIEF}}(g_{\text{right } k}))$ is the Haussdorff distance of the two BRIEF features f_{BRIEF} of the glints $g_{\text{left } l}$ and $g_{\text{right } k}$. The scaling parameter σ_{BRIEF} defines the tolerance of deviation in the BRIEF features.

Considering the planar alignment of the cameras and the fixed distance between them, the angle between g_{left}, and the corresponding g_{right} can be deduced. However, taking into account any imperfect distortion or alignment of the cameras, the geometric constraints are also described as an unnormalized distribution to accommodate minor deviations:

$$v_{\text{angle } l,k} = g_{\text{right } k} \underset{xy}{\sphericalangle} g_{\text{left } l}, \tag{4.5}$$

and

$$\mathcal{L}_{\text{angle } l,k} = e^{-\frac{(C_1 \sphericalangle_{xy} C_0 - v_{\text{angle } l,k})^2}{2 \cdot \sigma^2_{\text{angle}}}}, \tag{4.6}$$

where C_1 and C_0 are matrices representing the intrinsic parameters of the corresponding cameras (see Figure 4.1.1). $\mathcal{L}_{\text{angle } l,k}$ provides the likelihood scaled by σ_{angle} of the angle between $g_{\text{left } l}$ and $g_{\text{right } k}$. Since $\mathcal{L}_{\text{angle } l,k}$ was designed to compensate for small errors in the alignment of the cameras, calibration, or other noise, it is recommended to choose $\sigma_{\text{angle}} \propto$ RMS Error of the intrinsic and extrinsic calibration.

Besides the angle of two detected glints in the stereo image, the minimal and maximal distance of the glints with respect to the working distance of w_{min} up to w_{max} is restricted as follows:

$$v_{\text{max}} = P_1 w^\top_{\text{min}} - P_0 w^\top_{\text{min}}, \tag{4.7}$$

and

$$v_{\text{min}} = P_1 w^\top_{\text{max}} - P_0 w^\top_{\text{max}}, \tag{4.8}$$

where P_0 and P_1 are projection matrices of both cameras. The vectors v_{min} and v_{max} describe the minimal and maximal pixel distance on the projection plane of both glints $g_{\text{left } l}$ and $g_{\text{right } k}$ inside the working distance. As above, the constraint is formulated as a likelihood function:

$$v_{\text{dist } l,k} = g_{\text{right } k} - g_{\text{left } l}, \tag{4.9}$$

and

$$\mathcal{L}_{\text{dist } l,k} = e^{-\frac{\|(0.5 \cdot v_{\text{min}} + 0.5 \cdot v_{\text{max}}) - v_{\text{dist } l,k}\|^2}{2 \cdot \sigma^2_{\text{dist}}}}, \tag{4.10}$$

whereby the scaling parameter σ_{dist} should be selected in such a way that v_{min} and v_{max} result in a likelihood above 0.5. Please note, it is assumed that one pixel on the sensor has the same scale in width and height. Otherwise, v_{dist} needs to be appropriately scaled.

The final matching likelihood $\mathcal{L}_{\text{match } l,k}$ of two glint candidates $g_{\text{left } l}$ and $g_{\text{right } k}$ is defined as follows:

$$\mathcal{L}_{\text{match } l,k} = \mathcal{L}_{\text{BRIEF } l,k} \cdot \mathcal{L}_{\text{angle } l,k} \cdot \mathcal{L}_{\text{dist } l,k}. \tag{4.11}$$

⑤ Triangulation Using triangulation, two corresponding glint image points are mapped from the respective camera coordinate system into 3D. In theory, it suffices to calculate the intersection of the two rays derived from the pinhole models of both cameras to obtain the projected point in 3D. Indeed, noise in the intrinsic and extrinsic calibration, as well as sampling inaccuracies, may lead to the fact that the derived rays probably will not intersect. The challenge is to find a 3D glint point that describes the detected corresponding glint image points optimal. There are several approaches to define optimality and how to calculate it [224], [225]. The most common and intuitive approach is to solve the linear equation system $P_0 \cdot g_{\text{scene}} = g_{\text{left}}$ and $P_1 \cdot g_{\text{scene}} = g_{\text{right}}$ for a least-squares error [226].

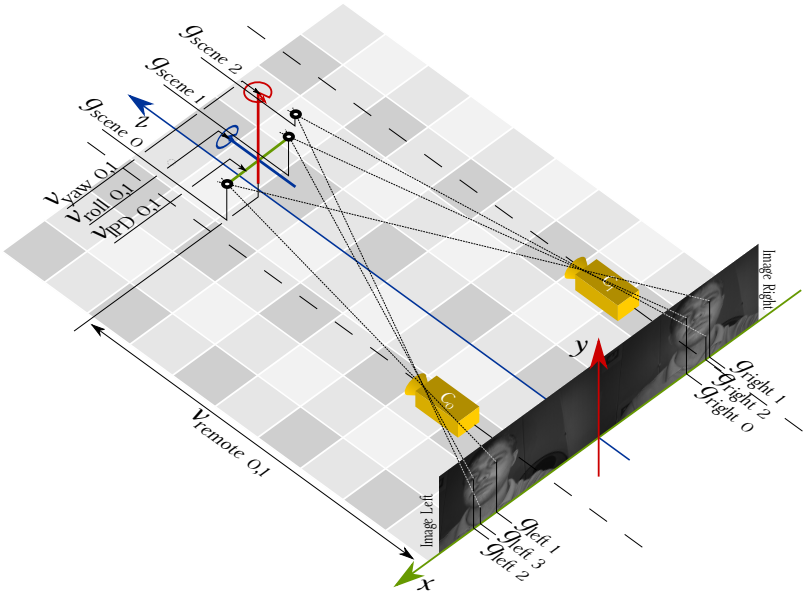


Fig. 4.1.1: After the triangulation of the matched glints in step ⑤, the pairwise likelihood $\mathcal{L}_{\text{pair}}$ of the 3D glint candidates is calculated according to Eq. (4.17), using the metrics v_{yaw} (vertical head rotation), v_{roll} (lateral head inclination), v_{IPD} (Interpupillary Distance) and v_{remote} (distance to the camera system).

⑥ **Stereo Glint Pair** After triangulating the detected glint candidates from the image points to 3D scene points; this step selects the most probable pair of glints representative to the left and right eye. Similar to step ④, a series of likelihood functions model the probability of a glint pair based on geometric constraints. This includes the Interpupillary Distance (IPD) $v_{\text{IPD }u,v}$, which is spanned by the two considered glint candidates $g_{\text{scene }u}$ and $g_{\text{scene }v}$, as well as the rotations $v_{\text{yaw }u,v}$ and $v_{\text{roll }u,v}$, and the distance $v_{\text{remote }u,v}$ relative to the camera system as shown in Figure 4.1.1.

The mean interpupillary distance on humans is around 63 mm, where the vast majority of adults have an IPD in the range 50 mm-75 mm [227]. Since both the IPD and the distance between the glints are directly linked to the distance between the eyeballs, it is further assumed that the distance of the glints $g_{\text{scene }u}$ and $g_{\text{scene }v}$ corresponds to the IPD:

$$\mathcal{L}_{\text{IPD }u,v} = e^{-\frac{(63\,\text{mm} - \|g_{\text{scene }u} - g_{\text{scene }v}\|)^2}{2\cdot\sigma_{\text{IPD}}^2}}, \tag{4.12}$$

where $\mathcal{L}_{\text{IPD }u,v}$ is the likelihood of the glint pair $g_{\text{scene }u}$ and $g_{\text{scene }v}$. It is recommended choose $\sigma_{\text{IPD}} \approx 11.04$ mm to keep $\mathcal{L}_{\text{IPD }u,v} \geq 0.5$ for the range 50 mm-75 mm for the IPD.

Apart from the IPD, rotation can be considered as an additional constraint for valid glint pairs. It is assumed that no valid glints can be expected if the vertical head rotation (yaw) exceeds a certain degree (i.e., $\pm 30°$). However, it cannot be ruled out that there are still head poses with a vertical rotation of more than $\pm 30°$ that still produce visible glints in both

eyes and both camera images. Hence a decreasing probability for such glints is assumed:

$$v_{\text{yaw } u,v} = g_{\text{scene } v} \underset{zx}{\sphericalangle} g_{\text{scene } u}, \tag{4.13}$$

and

$$\mathcal{L}_{\text{yaw } u,v} = e^{-\frac{v_{\text{yaw } u,v}{}^2}{2 \cdot \sigma_{\text{yaw}}^2}}, \tag{4.14}$$

where $\mathcal{L}_{\text{yaw } u,v}$ is the likelihood of the glint pair $g_{\text{scene } u}$ and $g_{\text{scene } v}$ with the vertical rotation $v_{\text{yaw } u,v}$. If no further details are known, it is recommended to choose $\sigma_{\text{yaw}} \approx 4.25°$ to keep $\mathcal{L}_{\text{yaw}} \geq 0.5$ for vertical head rotations less than $\pm 35°$. Simultaneously to the vertical head rotation, the lateral head inclination can be constrained by $\mathcal{L}_{\text{roll } u,v}$ and defined analogously to $\mathcal{L}_{\text{yaw } u,v}$.

Finally, the distance between the glint pair and the camera system is calculated, to purge glint pairs outside the defined working range w_{min} - w_{max}:

$$v_{\text{remote } u,v} = 0.5 \cdot g_{\text{scene } v} + 0.5 \cdot g_{\text{scene } u}, \tag{4.15}$$

and

$$\mathcal{L}_{\text{remote } u,v} = e^{-\frac{\| 0.5 \cdot w_{\text{min}} + 0.5 \cdot w_{\text{max}} - v_{\text{remote } u,v} \|^2}{2 \cdot \sigma_{\text{remote}}^2}}, \tag{4.16}$$

where $\mathcal{L}_{\text{remote } u,v}$ represents the likelihood of the distance of the glint pair $(g_{\text{scene } v}, g_{\text{scene } u})$ regarding the working distance. It is recommended to choose the scaling parameter σ_{remote} to obtain a likelihood of at least 0.5 can be obtained at the border of the working area. The overall likelihood $\mathcal{L}_{\text{pair } u,v}$ for a certain glint pair is defined as the product of all single metric likelihoods:

$$\mathcal{L}_{\text{pair } u_0,u_1} = \mathcal{L}_{\text{match } u_0,u_1} \cdot \mathcal{L}_{\text{match } v_0,v_1} \cdot \mathcal{L}_{\text{IPD}} \cdot \mathcal{L}_{\text{yaw } u,v} \cdot \mathcal{L}_{\text{roll } u,v} \cdot \mathcal{L}_{\text{remote} u,v} \tag{4.17}$$

where $\mathcal{L}_{\text{match } u_0,u_1}$ and $\mathcal{L}_{\text{match } v_0,v_1}$ are the matching likelihood of the two 2D glint candidates of the triangulated glint $g_{\text{scene } u}$ and $g_{\text{scene } v}$. The glint pairs with a likelihood higher than a certain threshold t (usually $t = 0.5$) are reported as valid detected glints:

$$d_{u,v} = \begin{cases} 0 & \text{if } \mathcal{L}_{\text{pair } u,v} < t \\ 1 & \text{otherwise} \end{cases}, \tag{4.18}$$

with $0 < t < 1$. $d_{u,v}$ indicates if the corresponding combination of glint candidates were classified as valid (1) or not (0).

Confidence measure $c_{u,v}$ indicates the certainty of a classification for a glint pair:

$$c_{u,v} = \begin{cases} \mathcal{L}_{\text{pair } u,v} & \text{if } \mathcal{L}_{\text{pair } u,v} > t \\ 1 - \mathcal{L}_{\text{pair } u,v} & \text{otherwise} \end{cases}. \tag{4.19}$$

Considering that the detection $d_{u,v} \in 0, 1$ of a glint pair in the stereo image is Bernoulli distributed, and the conjugated likelihood $\mathcal{L}_{\text{pair } u,v}$ is part of the exponential family, the prior probability for the correct classification can be modeled using a Beta distribution [228]:

$$P(\mathcal{L}_{\text{pair } u,v}) = \frac{{\mathcal{L}_{\text{pair } u,v}}^{\alpha-1} \cdot (1 - \mathcal{L}_{\text{pair } u,v})^{\beta-1}}{\mathrm{B}(\alpha, \beta)}, \tag{4.20}$$

where α and β control the expected correct and incorrect prediction rate, respectively. Given prior knowledge of the accuracy of the algorithm for a specific setup, it can be integrated into an advanced confidence measure $\hat{c}_{u,v}$:

$$\hat{c}_{u,v} = \begin{cases} P(\mathcal{L}_{\text{pair } u,v} < t) & \text{if } \mathcal{L}_{\text{pair } u,v} > t \\ P(\mathcal{L}_{\text{pair } u,v} > t) & \text{otherwise} \end{cases}. \tag{4.21}$$

Adaptive parameters The behavior of the presented classifier is strongly related to the choice of the parameters of each single likelihood function. The following describes the adaption of these parameters in an online fashion to adjust the classifier to the subject and to compensate for possible calibration or setup inaccuracies. Since the likelihood functions are considered as proportional to normal distributions (without constant normalization), the parameters of the likelihood function can be transformed to exponential distributions as well, in order to use Bayesian inference to update them after each iteration. Therefore, the likelihood $\mathcal{L}_{\text{pair } u,v}$ is redefined as the marginal likelihood:

$$\mathcal{L}_{\text{pair } u,v} = P(X_i|\eta_{i-1}) = \int_\theta P(X_i|\theta) \cdot P(\theta|\eta_{i-1}) d\theta, \tag{4.22}$$

where X_i is the considered set of glint candidates $\{g_{\text{left } u_0}, g_{\text{right } u_1}, g_{\text{left } v_0}, g_{\text{right } v_1}\}$ extracted in iteration i. $P(X_i|\theta)$ is the product of the likelihoods, as previously stated in equation 4.17 using the parameters θ. η_{i-1} is the set of the hyperparameters of the parameter distributions gain from the last iteration. The Bayes' theorem allows us to update the parameter distribution by the observed data as follows [229]:

$$P(\theta|\eta_i) \propto P(X_i|\theta) \cdot P(\theta|\eta_{i-1}) = P(\theta|\eta_0) \cdot \prod_{j=1}^{i} P(X_j|\theta), \tag{4.23}$$

where $P(\theta|\eta_0)$ is the probability of the parameters θ given the initial hyperparameters η_0, and $P(X_j|\theta)$ is the probability of the glint pair X_j given the parameters θ. This requires that the complete data series $X_0 \ldots X_i$ seen up to now must be stored. Since this is not practicable for large data series, the hyperparameters are estimated after each iteration anew:

$$P(\theta|\eta_i) = P(\theta|\eta_{i-1} + \delta\eta_i), \tag{4.24}$$

where $\delta\eta_i$ is the set of gradients between the hyperparameters η_i and η_{i-1}. All the likelihood functions $L_{\text{match}} \dots L_{\text{pair}}$ can be converted to the form:

$$\mathcal{L} = P(X_i|\{\mu, \sigma^2\} \in \theta) = e^{-\frac{(\mu - X_i)^2}{2 \cdot \sigma^2}}, \tag{4.25}$$

which corresponds to a non-standardized normal distribution. The likelihood of the parameters μ and σ are described by the distributions based on the hyperparameters η. The gradients $\delta\eta$ are approximated roughly by the distance of the observed data and estimated mean:

$$\begin{aligned} \delta\hat{\eta}_{\mu \text{ mean } i} &\approx \eta_{\mu \text{ mean } i-1} - X_i, & \delta\hat{\eta}_{\mu \text{ variance } i} &\approx \delta\hat{\eta}_{\mu \text{ mean } i}{}^2, \\ \delta\hat{\eta}_{\sigma \text{ mean } i} &\approx \eta_{\sigma \text{ mean } i-1} - (\eta_{\mu \text{ mean } i-1} - X_i)^2, & \delta\hat{\eta}_{\sigma \text{ variance } i} &\approx \delta\hat{\eta}^2_{\sigma \text{ mean } i}, \end{aligned} \tag{4.26}$$

and normalized by a weight function h (e.g., tanh) and the marginal likelihood:

$$\delta\eta_i = h(\delta\hat{\eta}_i) \cdot P(X_i|\eta_{i-1}). \tag{4.27}$$

Finally, η_i is estimated using Stochastic Gradient Descent (SGD) to minimize $\delta\hat{\eta}$ [230]. This way, the parameters of the model are continuously fitted to the observed glints. This works fine for parameters modeling fixed physical circumstances, such as the IPD. However, when adapting temporal parameters, e.g., the distance between the glints and the camera system, the system must be able to adapt to the new situation continuously. Otherwise, it may happen that after some time of recording and adaption, the model overfits and is not able to detect the glints anymore (e.g., if the subject moves around). Therefore, all the hyperparameters representing the variance are limited to a lower bound. As a fallback, the model generalizes itself when the confidence level falls below a critical value $t_{\hat{c}}$. Thus, the model is able again to adapt to larger changes after a few iterations:

$$\delta\eta_i = \begin{cases} h(\delta\hat{\eta}_i) \cdot P(X_i|\eta_{i-1}) & \text{if } \hat{c} > t_{\hat{c}} \\ h(\eta_0 - \eta_{i-1}) & \text{otherwise} \end{cases}. \tag{4.28}$$

The number of extracted glint candidates per frame and the resulting number of possible combinations of glints has a considerable impact on the runtime. Hence, the threshold t_{fast2} is adapted with the gradient $\delta\hat{\eta}_{\text{fast2}}$ as the difference between the number of detected glints and the number of expected glints. Moreover, the ROI (defined by a center point and the width and height of a rectangle: $R = (x, y, w, h)$) is adapted to reduce the search area for glint candidates. The gradient of the ROI ($\delta\hat{\eta}_{\text{ROI}}$) is calculated as the difference between the ROI from the last iteration and an ROI defined around the detected glint with fixed padding.

Table A.1 in the appendix provides a comprehensive overview of all optimized hyperparameters η and a recommendation for the respective initial state η_0.

Implementation The process presented here has been integrated into Caffe [231]. The steps ②-⑥ are implemented as custom layers. Thus, during the forward propagation, the Glints candidates are extracted, and their probabilities are calculated. The backpropagation

is used to calculate the gradient $\delta\eta$, and subsequently to optimize the hyperparameters η using Caffe's built-in SGD solver. Besides the layers to implement steps ②-⑥, additional layers to calculate the confidence (also used as a loss), evaluation, debugging, and writing of journal files were added.

The division of the individual steps into different Caffe layers allows a simple replacement through other approaches or the integration of the individual steps into a deep learning approach. Besides the feature extraction outlined in step ② to highlight glint candidates, various additional state-of-the-art methods are implemented, which have been reported in the literature: (Adaptive) Threshold, Ebisawa, Canny, LoG, and Sobel. Some of these methods were slightly modified in order to optimize their behavior by adapting their parameters in the backpropagation All extractors were extended through a spatial normalization of the output, to emphasize glints located in a relatively low light area more strongly. This significantly increases the detection rate of low illuminated glints, yet also increases the noise.

4.1.2 Evaluation

The above glint detection approach was evaluated on an extensive hand-labeled dataset. More specifically, the performance of various glint extractors from the state-of-the-art is compared to the proposed FAST2 method (see appendix Figure A.1). Moreover, the approach was benchmarked to two state-of-the-art methods for the detection of faces and eyes (see Table 4.1).

Dataset The dataset consists of $6,993$ hand-labeled IR stereo images (1152 px $\times$ 1536 px at 10 Hz). The images were recorded by two planar arranged cameras with a vertical and horizontal offset to each other of 150 mm and 20 mm. Overall, the dataset provides seven sequences (DS01, …, DS07), where each sequence consists of 999 stereo frames (99.9 s) collected from four different subjects. All sequences, except for DS0, contain at least one of four different types of lighting interference, which makes glint detection on these images very challenging. Figure 4.1.3 and 4.1.2 provide an example of each challenge and an overview on which of those are present in the respective sequences.

The sequence DS01 is considered the easiest one since it presents a favorable setting for glint detection. In contrast, the sequences DS02, DS04, and DS05 contain strong illumination gradients due to sunshine, which additionally causes reflections on hair, jewelry, buttons, and other reflecting parts in the scene. In DS02, DS03, and DS05 additional reflecting objects were placed in the scene, which would cause additional disturbances and possibly even cover the actual glints. In the sequences DS06 and DS07, the subjects wear sunglasses. This causes additional reflection around the eye, which sometimes overlay the original glint. DS05 provides the most challenging lighting conditions since a strong infrared illumination source was moving around the subject.

The intrinsic parameters were estimated using Matlab's camera calibration toolbox and a 10×8 checkerboard. All visible glints in the stereo images were labeled and triangulated, thus the glint position is available in 2D image coordinates as well as 3D coordinates of the stereo camera.

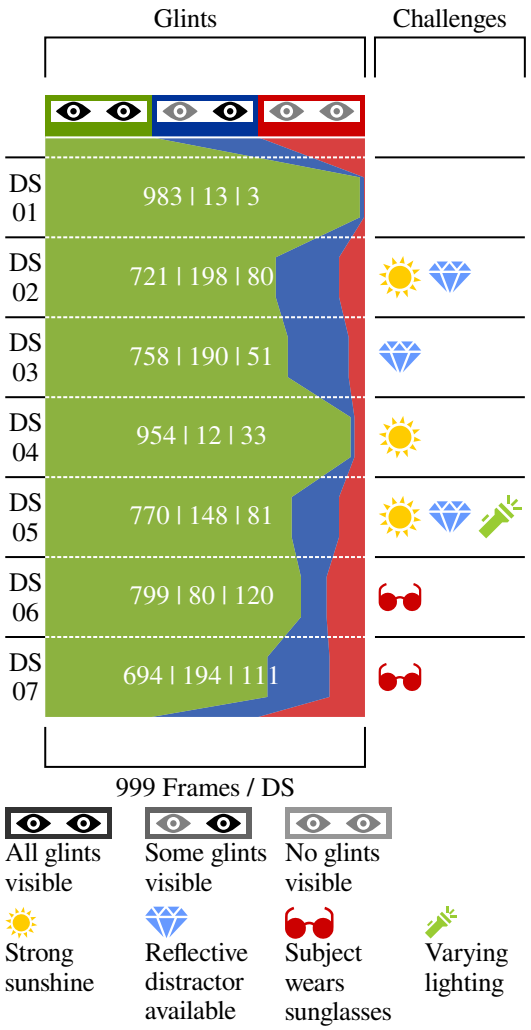
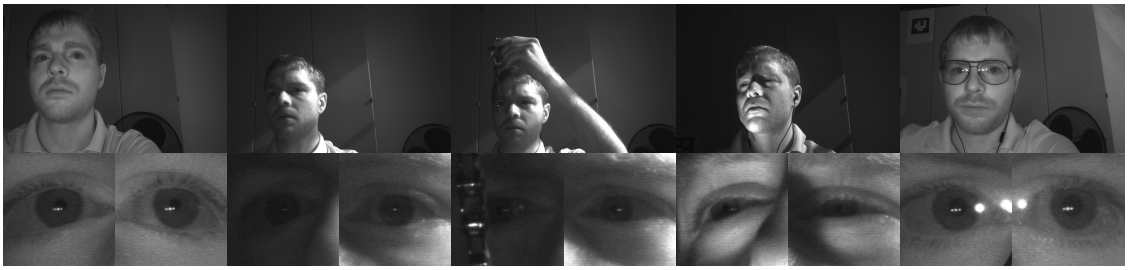


Fig. 4.1.2: Overview of the challenges covered by the presented dataset. The first dataset DS01 is considered as easy challenge. It provides constant illumination without distractors in the scene. Apart a few exceptions, glints are continuously available in both eyes. The data sets DS02 to DS05 comprise additional challenges. Either distractors, strong sunlight, or both are provided. DS05 additionally includes a strongly varying light source. The subjects in DS06 and DS07 wear sunglasses, causing additional reflection on the lenses and frame. An example of each challenge type is provided in Figure 4.1.3. Each dataset provides 999 sequential stereo images (total $2 \times 6,993$ labeled images).

Glint extractors The appendix Figure A.1 shows the evaluation performance of the proposed glint detection using different methods from the state-of-the-art (Adaptive Threshold, Canny, Ebisawa, LoG Sobel) and the proposed FAST2 for feature extraction. Except for the Canny edge detector, all methods achieve predominantly satisfying performance. The proposed extractor FAST2 is characterized by an overall very low false-positive rate, which can be traced back to the implicit limitation of the size of possible glints (r). The dataset DS01 was detected almost without errors. The challenging lighting conditions in data sets DS02, DS04, and DS05 do not appear to have a significant impact on the performance of the presented algorithm. In data set DS07, the user's sunglasses induce additional reflections along with the frame and on the lenses. This leads to a large number of detected glint candidates around the eyes. Thus, the actual glint is suppressed due to the distance condition to the previously selected glints in step ③, which finally reflects in an overall higher false-positive rate.

Eye detection Table 4.1 shows the results of a comparison between the proposed method based on the FAST2 feature extraction and OpenFace [232] (for facial recognition) and a



Good illumination | Strong sunshine | Distractors | Illumination | Sunglasses

Fig. 4.1.3: The dataset comes with various challenges. The first frame shows an example of favorable conditions for glint detection. The next image contains a huge light gradient caused by lateral sun radiation. Thereby, the intensity of the right glint is suppressed. In the third frame, an additional distractor in front of the right eye causes additional reflections next to the glint. The 4th example shows the impact of a further varying IR light source. In the last frame the subject wears sunglasses. IR is not blocked by the glasses so the glints are well visible, but causes additional reflexions close to the eye.

	OpenFace		FAST2		HC	
	Accuracy	F_1 score	Accuracy	F_1 score	Accuracy	F_1 score
DS01	0.862	0.926	**0.989**	**0.995**	0.446	0.617
DS02	**0.642**	**0.782**	0.609	0.757	0.050	0.094
DS03	0.586	0.739	**0.834**	**0.910**	0.499	0.666
DS04	**0.987**	**0.994**	0.887	0.940	0.026	0.051
DS05	0.697	0.822	**0.809**	**0.894**	0.086	0.158
DS06	**0.946**	**0.972**	0.901	0.948	0.064	0.120
DS07	**0.724**	**0.840**	0.716	0.835	0.019	0.037
AVG	0.777	0.869	**0.821**	**0.897**	0.170	0.256
All	0.790	0.883	**0.833**	**0.909**	0.177	0.301

Accuracy | F_1 score

Tab. 4.1: Evaluation results of our approach used as an eye detector compared to the two state of the art methods OpenFace and HC.

cross-validated HC. OpenFace and the HC were applied separately on both stereo images and determined the center of the eye. Next, these points were triangulated and compared to the labeled points from the dataset. Since the labeled glint points do not necessarily coincide with the center of the eye, an error up to 2 cm was considered as correct detection. Moreover, the dataset was cleaned from all images with visible eyes, but no corresponding glints.

The presented method achieved performance similar to that of OpenFace and partly outperformed it. The performance of OpenFace collapses, especially on the dataset DS03 and DS05. For these datasets, the tracking in OpenFace partially interrupts; for DS03, the head of the subject is not always completely visible in both stereo images, whereas for DS05, the performance decreases due to varying illumination. On average, the presented method achieves an accuracy of 82.1 % and 83.3 % over the whole dataset. However, the HC shows inferior performance, possibly due to the unique lighting conditions in each dataset and the susceptibility of Haar-Like features to inconsistent lighting [233]. Furthermore, the detected bounding boxes are often inaccurate; thus, the maximum accepted error of 2 cm is exceeded after triangulation. Only two data sets, namely DS01 and DS03, which contained neither challenging lighting conditions nor sunglasses, were partially detected correctly.

4.1.3 Limitations

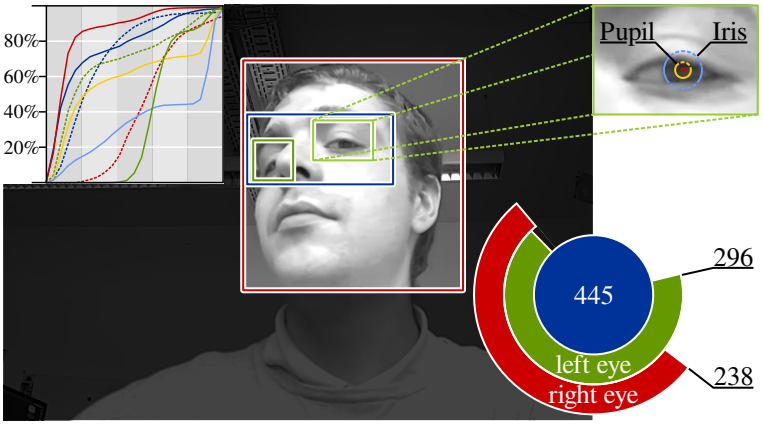
The presented approach intends to counteract the limitations of state-of-the-art glint detectors towards a more robust and particularly illumination interference-resistant glint detection. However, the additional filter step requires a stereo camera or at least a depth sensor for operation. Although given in the most state-of-the-art remote eye-tracking systems, some mono camera systems (e.g., Eye Tribe) may not benefit from this approach.

4.1.4 Conclusion

This section introduced an adaptive glint detection approach for remote eye-tracking in the wild based on a probabilistic model. The evaluation extensive and challenging dataset reveals that the proposed algorithm significantly improves the state-of-the-art and copes with various challenges in real-world settings, such as varying illumination, reflections, or distractors. Thus, the reported 3D glint positions, along with the profound confidence measure and the 5 degrees of freedom head-pose, provided a strong basis to parametrize eye models. Moreover, a new dataset consisting of $13,986$ hand-labeled images provides a solid foundation to benchmark new glint and eye detectors.

The following section addresses the next step in the remote eye-tracking stack. After the successful detection of the eye-region and head-pose, a pupil detection algorithm reveals the last required puzzle piece to processes the user's line of sight.

4.2 Evaluation of state-of-the-art pupil detection for remote eye-tracking



Pupil detection is a crucial yet particularly challenging step in the remote eye-tracking stack. Even though cameras in remote eye-tracking usually offer higher resolutions, the actual eye's resolution is usually lower than in head-mounted eye-tracking. Simultaneously, the eye's contrast is significantly lower, posing particular challenges to edge-based pupil detection algorithms. These ample hurdles are doubtless, one reason for the small set of publicly available pupil detection algorithms designed for remote eye-tracking systems, and applicable across multiple application domains and technical setups.

This section evaluates the performance of several pupil detection algorithms from the domain of remote as well as head-mounted eye-tracking across various distinct datasets. The evaluation involves two paradigms: First, the evaluation of the pupil detection rate and accuracy based on manually labeled eye images. This evaluation reflects the actual detection rate, excluding the impact of the preceding steps in the remote eye-tracking stack. The second scenario investigates the performance based on detected eye regions. Thus the evaluation includes the impact of, e.g., varying size, shape, and position of the eye images.

The following sections are supported by a prior publication from Fuhl & Geisler et al. at the Conference on Pervasive and Ubiquitous Computing (UBICOMP) in 2016 [13]. Section 4.2.1 elaborates the used benchmark datasets, including the new dataset provided by Fuhl & Geisler et al. Moreover, this section introduces the preparation of the two evaluation paradigms: based on labeled or detected eye regions. Section 4.2.2 provides an extensive evaluation and benchmark of the pupil center detection of state-of-the-art algorithms from the domain of remote eye-tracking, as well as head-mounted eye-tracking. The Sections 4.2.3 and 4.2.4 address the presented approach's limitations and conclude this section.

4.2.1 Method

The evaluation compromises 5 state-of-the-art pupil detection algorithms, which were initially designed for head-mounted eye-tracking and high-resolution eye images: Świrksi [60], Starburst [59], ExCuSe [62], SET [61], and ElSe [63]. All 5 algorithms base on edge filters. Yet, ExCuSe and ElSe provide a fallback mechanism in case the edge-based approach fails. All algorithms are elaborated in detail in Section 2.2.2, along with a performance evaluation on head-mounted eye-tracking eye images.

Moreover, 3 algorithms were evaluated, which were specially designed for the application in remote eye tracking: George and Routray [97], Droege and Paulus [94], and Timm and Barth [95]. All 3 algorithms base on gradients angles instead of edges. Section 2.3.3 provides a detailed introduction to these algorithms.

Dataset The pupil detection rate of each algorithm is evaluated on 3 extensive data sets, namely: BioID [234], GI4E [235], and a newly created dataset by Fuhl & Geisler et al. (hereinafter referred to as *New dataset*) [13].

BioID: The BioID dataset provides $1,521$ grayscale images (384 px $\times$ 286 px) showing faces of 23 subjects recorded at different office environments with varying illumination. Included challenges are different camera distances to the subjects, blinks, and small head movements. Figure 4.2.2 provides 4 random example images from the BioID dataset. The dataset was collected to evaluate face detection and recognition algorithms. Hence, the original BioID provides only labels of eye center, not of the pupil center. Yet, there are several further labels (pupil center and facial landmarks) available for this dataset provided by the FGnet project [234], [236].

GI4E: The GI4E dataset comprises $1,236$ RGB images (800 px $\times$ 600 px) of 103 subjects recorded by a standard webcam. The images show a frontal face view of the subject. For each subject provides 12 sequential frames, where the subject gazes at different points on a screen. Thus the subject's face is aligned straight to the camera center. The data set does not involve any particular challenge, except for some subjects wear glasses. Nevertheless, due to the high image resolution, the constant illumination, and the well aligned-camera perspective, this dataset is considered as an easy challenge for remote pupil detection. Figure 4.2.2 presents four random example images of the GI4E dataset. The dataset contains labels of the eye corners and the iris center. However, to evaluate the pupil detection rate, the iris center was considered to be the same as the pupil center [235].

New dataset: In their publication, Fuhl & Geisler et al. also provided a new recorded dataset, consisting of 445 manually-labeled images of two subjects (Fuhl & Geisler). This dataset was recorded in an office environment by a FOSCAM FL9826P near-infrared surveillance camera. The dataset contains distinct head rotations, eye movements, blinks, reflections, and pronounced light gradients. The dataset consists of near-infrared grayscale images with a 1280 px $\times$ 960 px resolution. Each frame provides labels of the iris contour, pupil contour, and the bounding boxes around

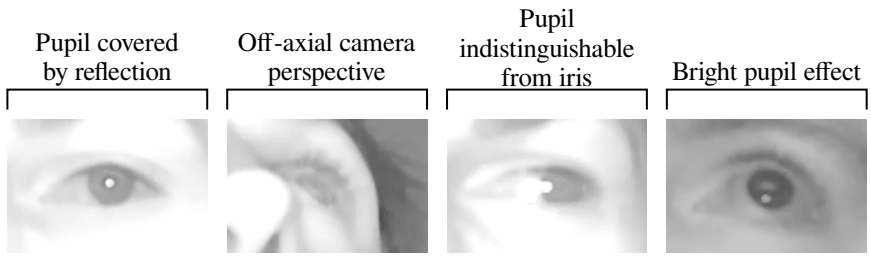


Fig. 4.2.1: Challenges for pupil detection provided by the new dataset.

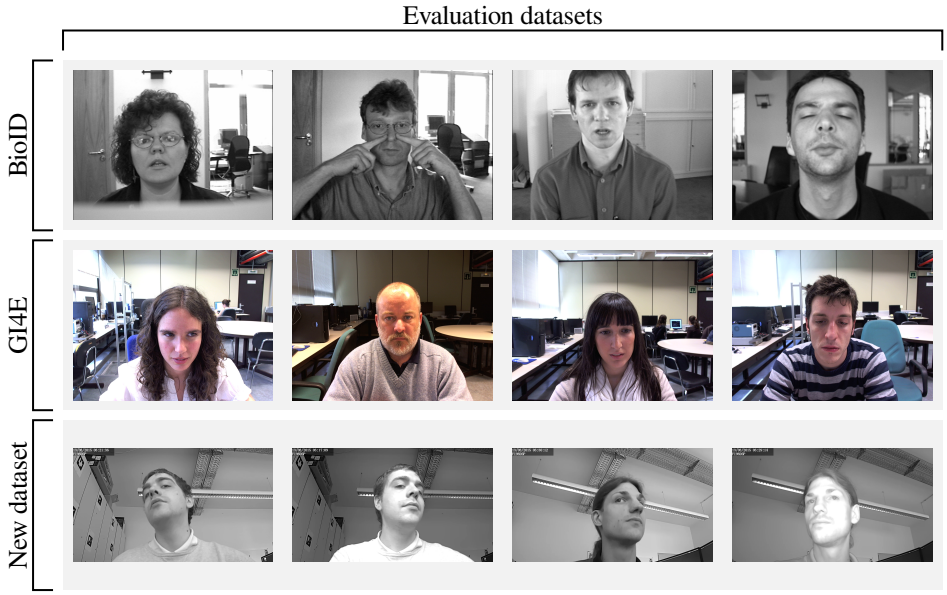


Fig. 4.2.2: Examples images from the evaluation datasets.

the eyes [13]. Figure 4.2.1 shows some example images from this dataset. The dataset was intended to be particularly challenging for pupil detection algorithms. Figure 4.2.2 shows some example eye images of some of the included challenges for pupil detection.

Eye region Detecting eye regions in remote data is a non-trivial task and usually produces eye regions with varying shape, size, and position. The datasets were further prepared to two scenarios, to examine the impact of such effects:

Manually labeled eye regions: The eye regions were cropped using manually created labels. In the case of BioID and GI4E, the eye region was determined by the labeled facial landmarks' respective eye corners. The New dataset provides eye bounding boxes; thus, the eye region can be directly cropped.

HC detected eye regions: The eye region was detected by a hierarchical arrangement

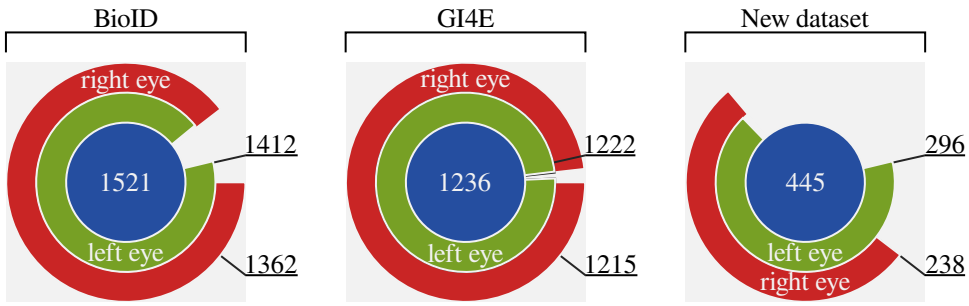


Fig. 4.2.3: The proportion of eye regions correctly detected by HC and KLT feature tracking.

of pre-trained HCs combined with a Kanade-Lucas-Tomasi (KLT) feature tracking using the implementation provided by OpenCV [237]–[239]. All correct detected eye regions (the pupil is inside the bounding box) were further applied to crop out the eye images. Figure 4.2.3 shows the number of resulting eye images over the three datasets. The HC and KLT feature tracking generated $2,774$ successful detections on the BioId data set (91 %), $2,437$ on the GI4E (99 %), and 534 on the new data set (60 %). In contrast to the BioID and GI4E data sets, the new data set includes large head rotations and occluded eyes (see Figure 4.2.2), which significantly reduces the number of successfully detected eye regions.

Table 4.2 shows the minimum, maximum average, and median size of the cropped eye images. The strong variation of the eye bounding box size during automatic detection, and the low resolution of the eye region, even with high-resolution images from GI4E or the New Data Set, is remarkable.

4.2.2 Evaluation

The cropped eye images were subsequently processed by the above-listed pupil detection algorithms to evaluate their detection rate. The best parameters for each algorithm were explored based on the performance achieved on the BioID data set. These parameters were subsequently used for further evaluation.

Figure 4.2.4 shows the evaluated algorithm's pupil detection rate depending on the euclidean distance between the predicted and labeled pupil center. The first two columns provide the detection rate depending on an absolute error (pixels), whereas the second two columns provide the detection rate depending on a relative error (divided by the eye image diagonal size). Thus, effects arising from different eye image sizes are compensated. Table 4.3 provides the exact pupil detection rate at a relative error tolerance of $\leq 20\,\%$ of the image diagonal.

In most cases, the algorithms' pupil detection rate benefits from automatically detected eye regions. This is mainly since the eye region detection, and pupil detection fail for similar challenges. In other words, eye region detection filters, many particularly challenging cases for pupil detection. More importantly, the evaluation reveals that algorithms intended for

BioID	Manually labeled	HC detected
Minimum	20.0 px × 40.0 px	12.0 px × 18.0 px
Maximum	20.0 px × 40.0 px	35.0 px × 52.0 px
Mean	20.0 px × 40.0 px	21.2 px × 31.8 px
Median	20.0 px × 40.0 px	21.0 px × 32.0 px

GI4E	Manually labeled	HC detected
Minimum	20.5 px × 20.9 px	26.0 px × 39.0 px
Maximum	26.7 px × 44.9 px	42.0 px × 62.0 px
Mean	22.0 px × 31.2 px	30.1 px × 45.2 px
Median	21.5 px × 30.7 px	30.0 px × 45.0 px

New dataset	Manually labeled	HC detected
Minimum	22.0 px × 30.0 px	35.1 px × 49.7 px
Maximum	47.0 px × 99.0 px	79.0 px × 118.0 px
Mean	24.5 px × 60.8 px	49.8 px × 74.5 px
Median	22.0 px × 61.0 px	49.0 px × 74.0 px

Tab. 4.2: The minimum, maximum, mean, and median resolutions of the labeled and detected eye region in the evaluation datasets.

	BioID		GI4E		New dataset	
	Manually labeled	HC detected	Manually labeled	HC detected	Manually labeled	HC detected
GeRo	77.2 %	88.4 %	79.4 %	76.0 %	47.9 %	70.3 %
DrPa	9.6 %	73.9 %	30.4 %	28.3 %	6.6 %	31.6 %
TiBa	69.6 %	82.3 %	80.6 %	86.7 %	89.1 %	83.8 %
SET	59.9 %	63.8 %	18.9 %	39.5 %	28.8 %	32.3 %
Starburst	27.3 %	91.5 %	90.4 %	83.5 %	6.6 %	80.2 %
Świrski	75.9 %	79.9 %	72.9 %	81.5 %	60.5 %	81.4 %
ElSe	90.7 %	93.9 %	98.3 %	89.8 %	92.7 %	93.3 %
ExCuSe	1.1 %	79.6 %	48.4 %	31.1 %	3.7 %	6.5 %

Tab. 4.3: The evaluated algorithms' detection rate at a relative error tolerance of $\leq 20\,\%$.

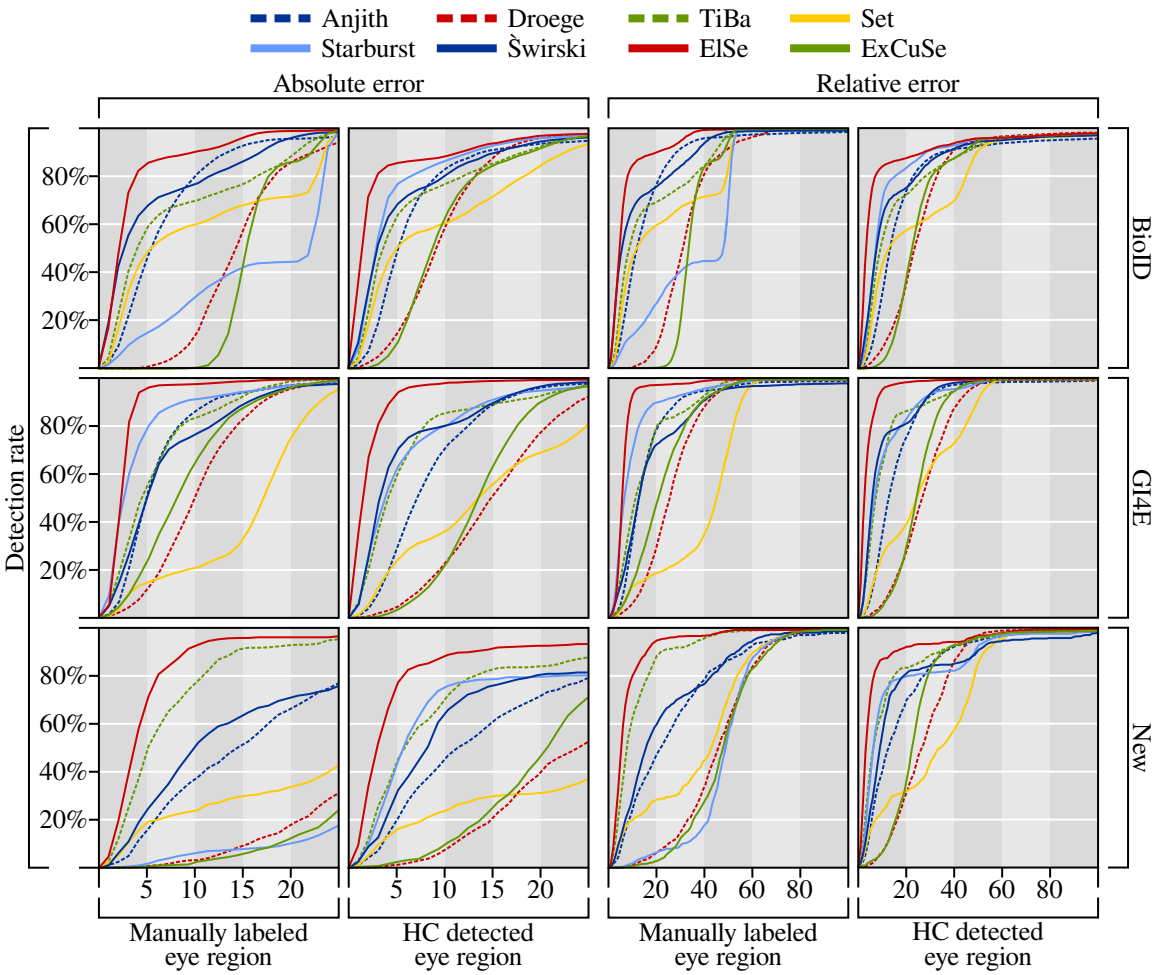


Fig. 4.2.4: The pupil detection rate of the evaluated algorithms depending on a maximum tolerated error. The relative error refers to the diagonal of the eye image.

remote image pupil detection do not necessarily achieve better detection rates than algorithms intended for head-mounted eye-tracking. However, amid the evaluated approaches, the algorithm ElSe [63] and the approach by Timm and Barth [95] exhibit a stable detection rate on all three datasets, which hints that these algorithms are robust against various challenges, e.g., difficult illumination or off-axial camera position.

The overall superiority of ElSe's detection rate compared to the other algorithms is mainly based on its fallback mechanism. If no suitable edges for an ellipse fitting are found, ElSe uses a convolutional filter chain, similar to a blob detection, to estimate the coarse the pupil. This, as a fallback intended coarse detection of the pupil center, reveals as very powerful in remote eye-tracking.

4.2.3 Limitations

Detecting the pupil in remote eye images is a challenging task. Conventional edge-based approaches fail since the image contrast is too low, and edges are not sufficiently pronounced. However, even gradient-based pupil detection algorithms explicitly intended to run on low-resolution and low-contrast eye images, show only in the instance of Timm and Barth sufficient detection rates. ElSe, a pupil detection algorithm designed for eye images of a head-mounted eye-tracker demonstrated the highest performance with a $\sim$90 % detection rate across all data sets.

However, the considered $\leq$ 20 % error tolerance in Table 4.3 corresponds to an average error tolerance of $\sim$8.9 px, $\sim$7.6 px, and $\sim$13.1 px for the labeled eye regions of the respective BioID, GI4e, and new data sets. For the detected eye region, $\leq$ 20 % deviation corresponds to an average error tolerance of $\sim$7.6 px, $\sim$10.8 px, and $\sim$17.9 px. Compared to the results of head-mounted eye-tracking (Section 2.2.2), it emerges that remote eye-tracking stack deals significantly lower accuracy in the pupil detection.

4.2.4 Conclusion

Pupil detection in remote eye-tracking deals with particularly tough challenges of low-resolution images, low image contrasts, unfavorable illumination, and off-axial camera perspectives. Only a few state-of-the-art algorithms address these points.

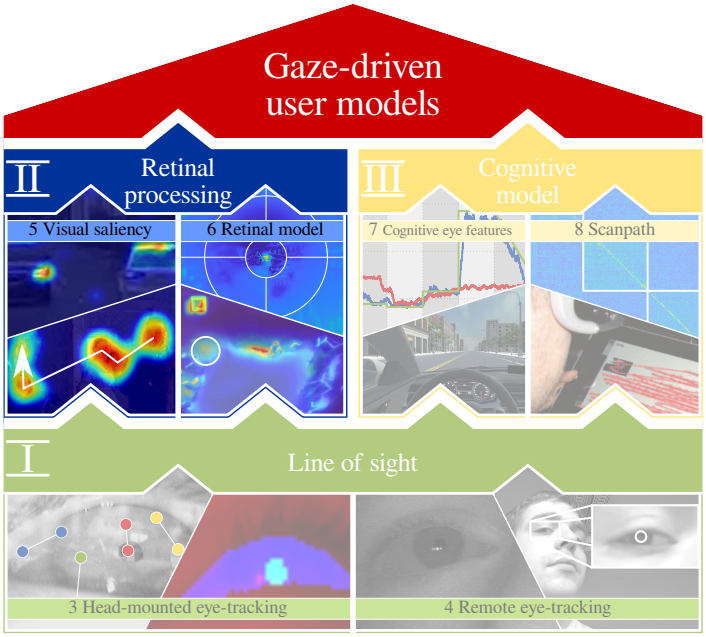
In this section, 8 state-of-the-art pupil detection algorithms were evaluated regarding their performance in pupil center detection on remote eye images. Three of them were specifically designed to run on remote eye images. Five further algorithms were borrowed from the domain of head-mounted eye-tracking. In fact, the top performer, ElSe, is intended to run head-mounted eye-tracking eye images. Yet, its coarse fallback pupil detection achieved the highest detection rate. However, most edge-based methods suffer from the low pixel contrast in remote eye images.

The evaluation was conducted on the two widely used data sets BioID and GI4E. Furthermore, Fuhl & Geisler et al. prepared a particularly challenging data set to explore the limitations of pupil detection on remote eye images. The new dataset provides 445 sequential near-infrared remote images, including various challenges such as distinct head rotations, eye movements, blinks, reflections, and pronounced light gradients. Moreover, the dataset provides labels for pupil contour, iris contour, and eye bounding boxes. Thus, it is perfectly suited for remote pupil detection benchmarks. Yet, the dataset can also be used to evaluate eye region detection algorithms.

This part elaborated novel and state-of-the-art methods for eye image analysis and pupil detection. As introduced in Section 2.1, the extracted eye features may provide the fundamental for accurate and robust processing of the user's line of sight in 3D using either head-mounted or remote eye-tracking systems. However, the next Part II goes beyond the tracked line of sight. The user's line of sight reveals the scene's POR of foveal visual attention. Yet, human visual perception goes far beyond the foveal area. Part II elaborates perception models covering the stimulus processing over the whole retina – driven by the extracted line of sight.

PART II

Retinal processing



Retinal processing is the first step in human vision. How a user perceives a scene is therefore directly linked to the retinal visual stimulus processing and which signals are passed on to the visual cortex via the optic nerve. However, the analysis of the scene's visual stimulus is divided into two distinct paradigms: Saliency maps process the scene in terms to emphasize areas that stand out from their local and global neighborhood, and thus particular visual accessible for a user. These models work independently of a user and provide a generalized prediction of visual attention. However, in contrast to saliency maps, retinal perception models are driven by a user's line of sight and predict the visually perceivable scene content for a particular user in an online manner.

The following two chapters address both paradigms, including a novel approaches to utilize state-of-the-art saliency maps for a user-oriented prediction of the visual perception in Chapter 5, and two sophisticated retinal perception models, enabling accurate prediction of the user's perceived scene content in Chapter 6.

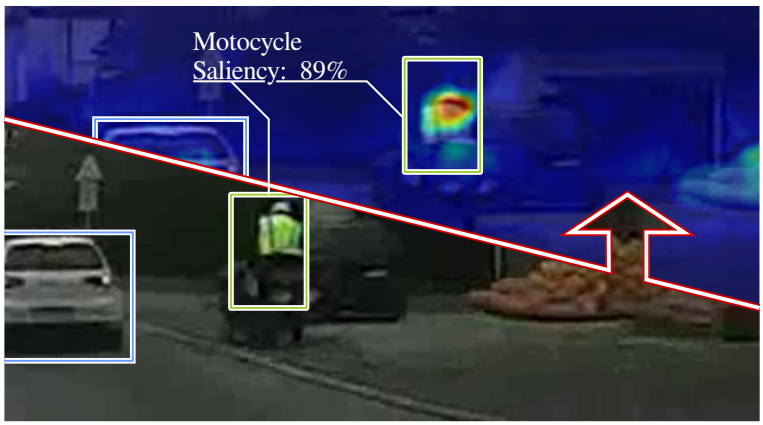
CHAPTER 5

Visual saliency

A complex reciprocal biological system of retinal feature extraction and cognitive attention selection swiftly aligns the user's line of sight to scene areas whose visual information is appreciated as most valuable in the immediate context. These can be scene areas that are related to the currently performed task, (e.g., traffic object while driving), or who may indicate possible hazardous situations (e.g., unexpected movements). While higher cognitive processes mainly drive the former, the latter is predominantly triggered by reflexes. However, both processes rely on the selection and interpretation of retinal extracted visual scene features.

Saliency models aim to replicate this retinal feature extraction to predict which scene area is likely to attract the user's visual attention. The following Section 5.1 introduces an extensive framework to enable researchers to develop novel saliency models by recombining state-of-the-art or implementing their own use-case specific approaches. Section 5.2 pushes the saliency models into a more user-oriented prediction by incorporating a tracked line of sight. Thus the presented joint model of visual saliency and eye-tracking enables prediction of the visual saliency from a user perspective.

5.1 Saliency sandbox



THE Saliency Sandbox provides a framework to develop and deploy state-of-the-art bottom-up saliency algorithms. The framework implements a runtime efficient yet flexible backend for quick and standardized prototyping, testing, and deployment of state-of-the-art and novel bottom-up saliency map approaches. Moreover, the Saliency Sandbox's Simulink S-Function wrappers enable deployments in a realtime signal processing environment, e.g., in the automotive context. Furthermore, Matlab MEX-Functions allow quick and runtime efficient prototyping using the full functionality of Matlab's runtime environment. The key advantages provided by the Saliency Sandbox are:

Research speed-up: Researchers may easily integrate their algorithms or employ out-of-the-box provided feature extractors and activation models to combine them into novel saliency maps. Interfaces for C++ and Matlab facilitate straight forward integration into existing toolboxes and systems.

Processing time: Written in C/C++, the framework backend accesses the full computational power of the target machine. However, for maximal parallelization, the GPU is accessible via Cuda and OpenCV, providing a significant speed-up through parallelization. Moreover, Simulink S-Functions enables realtime signal processing environments such as automotive or robotics.

Streaming: The Saliency Sandbox supports saliency maps on videos as well as on individual frames. Several features improve prediction accuracy on sequential data, e.g., optical flow, saliency tracking, or background subtraction as a new kind of feature map.

State-of-the-art saliency maps: The proposed framework comprises a large selection of pre-implemented feature extractors along with the most famous bottom-up activation models, e.g., Itti & Koch, Graph-based Visual Saliency, Frequency Tuned Saliency, Spectral Whitening Saliency, and Boolean Maps Saliency [240]–[244].

The following sections are supported by a prior publication from Geisler et al. at the Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP) in 2017 [6]. Section 5.1.1 provides the Saliency Sandbox's architecture. The out of the box provided feature extractors and activation models elaborated in Section 5.1.2. Section 5.1.3 demonstrates exemplary usage. The final Sections 5.1.4 and 5.1.5 state the limitations and conclusion.

5.1.1 Architecture

The vast majority of bottom-up saliency approaches implement a three-staged paradigm:

1st The extraction of feature maps. Feature maps may range from simple color channels to sophisticated scene features, e.g., depth, (semantic) segmentation, or motion.

2nd Processing of the (foveal retinal) activation. This step emphasizes salient regions in the beforehand extracted feature maps. While the feature extraction resembles most saliency approaches, the distinct activation steps comprise the core saliency algorithm.

3rd The final step normalizes, weights, and combines distinct feature map activations to a final saliency prediction.

Yet, the Saliency Sandbox supports this paradigm, but not limits to it. Rather, the process from a raw input image to a saliency map is modeled as a tree with processing steps as node structures that provide input and output. The root node usually combines several activation maps. Each activation, in turn, depends on a feature extractor, which may lead to further branches and nodes. Figure 5.1.1 illustrates an example of Saliency Sandbox's implementation of the Itti & Koch approach as a processing tree. Each node provides a uniform interface comprising an input and output buffer. This architecture allows an arbitrary combination of features and activations function. Thus, the Saliency Sandbox also allows experimenting in the hierarchy – e.g., by using a saliency map as activation.

Moreover, tree architectures provide the advantage that each branch is independently calculable. This enables significant runtime speed-up through parallelization, yet may cause redundant calculations of some features. Therefore, the Saliency Sandbox implements a producer-consumer model to limit the number of threads to an optimal amount regarding the CPU capabilities.

Each node implements the abstract base class *Algorithm* (see Figure 5.1.2). This abstract class provides basic functionalities such as input and output buffers, interfaces for Matlab and Simulink, and properties for parameterization. Algorithms processing images and videos can implement the specialization *ImageAlgorithm*, which already provides functions to convert RGB images to the appropriate input type and size, or utilities for loading and processing videos. The abstract methods *reset* and *calc* are called by the *Algorithm* base class and implement the actual algorithm. The method *reset* sets the algorithm to a default state, whereas *calc* calculates the output based on the input and parameters.

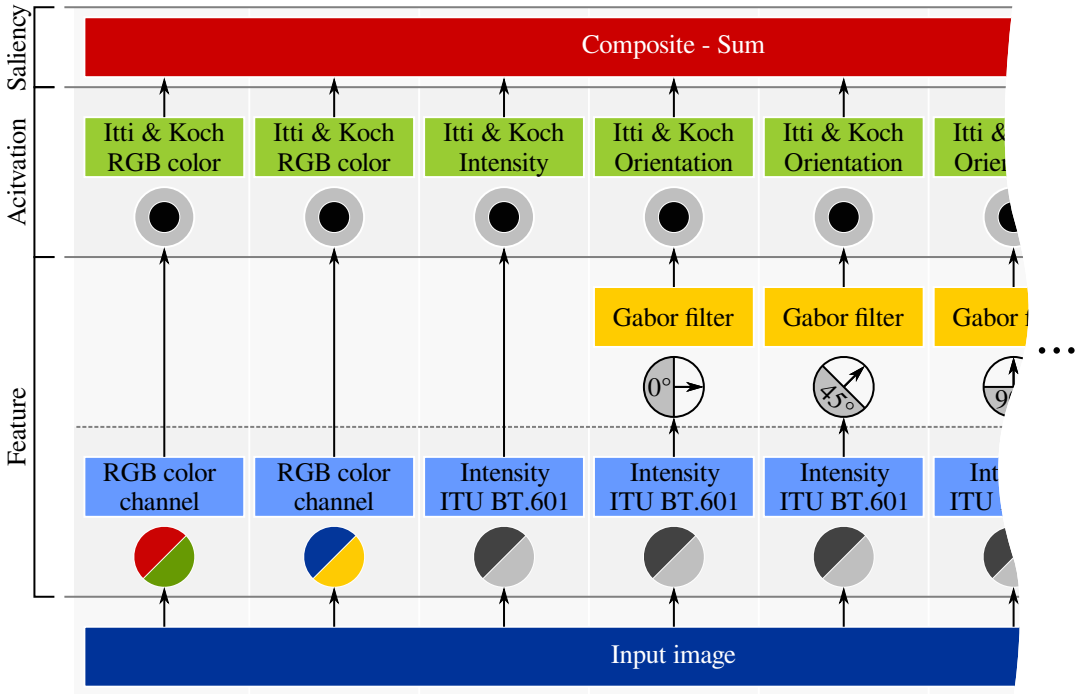


Fig. 5.1.1: Implementation of the Saliency Sandbox's Itti & Koch model. The activation is calculated by center-surround differences using pyramids over seven scales and normalized by iteratively $\mathcal{DoG}$ filtering, followed by a linear rectification. All features are extracted from the RGB colorspace. The RGB color channel feature extracts an individual channel from the RGB colorspace or, in this case, the pseudo channel as differences of red and green, respectively, blue and yellow. The intensity is calculated in the standard ITU BT.601 [245]. Gabor filters extract the orientation features with different rotations (0°, 45°, 90°, 135°). A weighted sum realizes the composition of the activations maps [243].

Besides the interfaces to Matlab and Simulink, algorithms from the Saliency Sandbox can also be integrated into existing projects as a library or executed as a standalone tool. For the latter, the base class *Algorithm* and the *ImageAlgorithm* class provide methods to, e.g., parse program arguments as algorithm properties, or to load input data such as images, videos, or masks.

Using the provided CMake-based build system, the compiled binaries include all the dependencies to OpenCV 2.4 and Boost, to enable portability along with different systems. Thus, new libraries and programs can be created without modification of the build scripts.

5.1.2 Algorithms

The Saliency sandbox provides a broad set of pre-defined features and activation maps. Most of them implemented by extensive use of the OpenCV 2.4 library for image processing and CPU handling. Figure 5.1.3 provides a comprehensive overview of the out-of-the-box

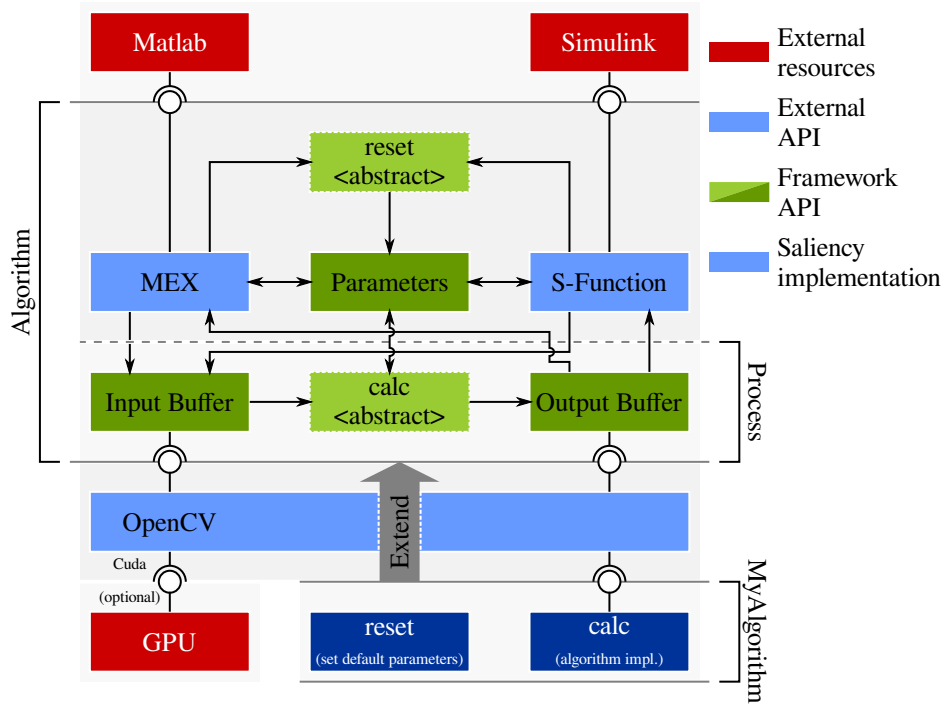


Fig. 5.1.2: The abstract *Algorithm* base class provides interfaces for Matlab and Simulink and implements the buffers for input and output. Furthermore, this class provides a property handling for parameterizable algorithms. The buffers are described using OpenCV data types. The integration of OpenCV allows access to a wide range of existing image processing functions and easy access to the GPU. New algorithms just have to implement the abstract functions *reset* and *calc* to provide new functionalities to the *Saliency Sandbox*.

available feature and activation maps.

The subsequent three sections elaborate each implemented algorithm analogous to the 3-phase paradigm. Furthermore, an example output of a driving scene accompanies each algorithm presented. Figure 5.1.4 shows the therefore used input stream, recorded by an action cam mounted behind the windshield. Driving scenes pose particular challenges, especially for motion-based features, since the car's ego-motion causes an omnipresent, yet varying flow in the image.

Features

The feature extraction divides into three categories, as illustrated in Figure 5.1.3: 1st) color-based features. These features extract distinct color channels of various color spaces (Grayscale, RGB, LMS, DKL, LAB). Moreover, inspired by the retinal opponent color pathways, the features may create contrasts between complementary color channels. 2nd) frequency-based features provided by an orientational Gabor filter. 3rd) Motion features for dynamic scene images, e.g., optical flow, background subtraction, and flicker.

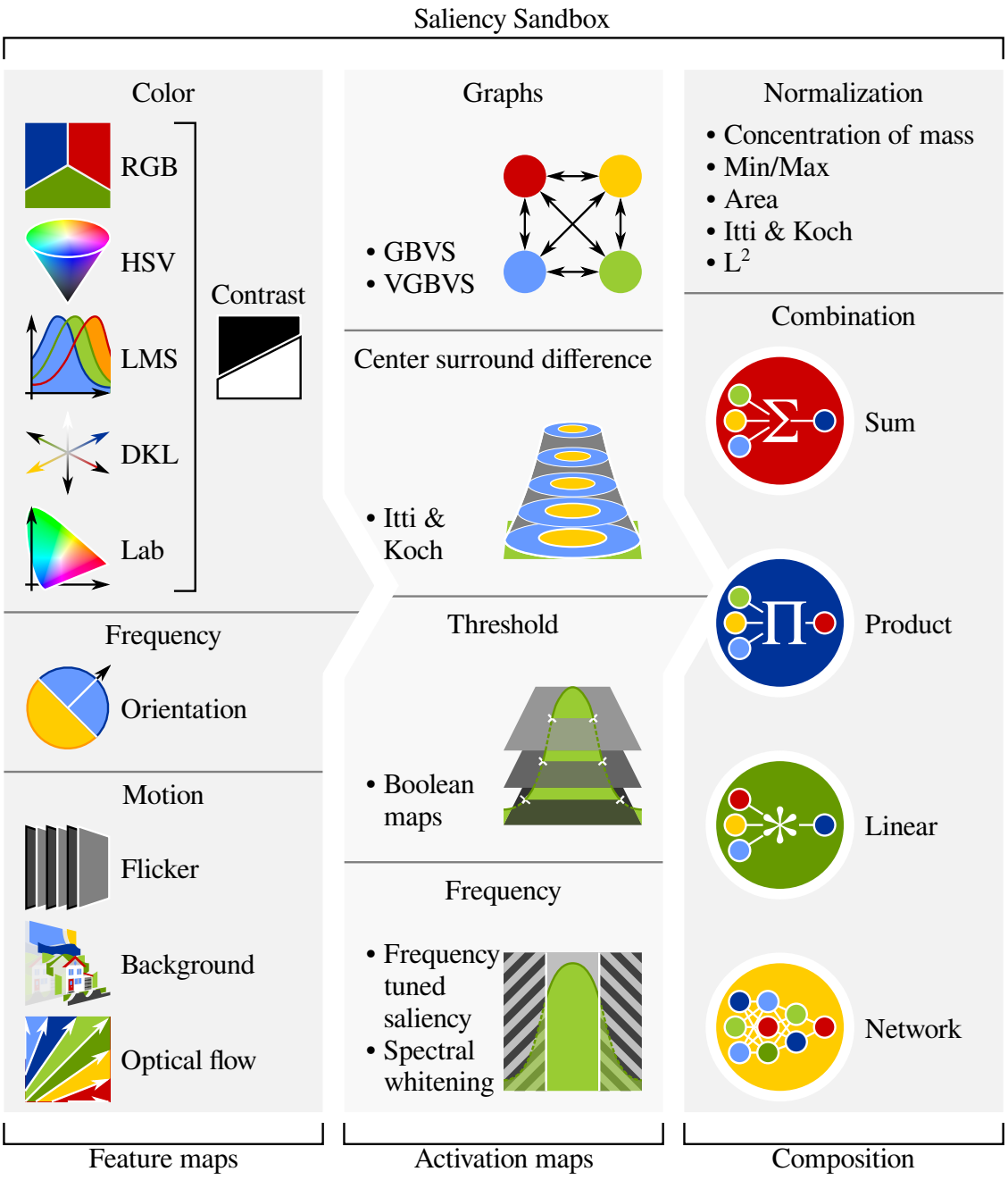


Fig. 5.1.3: The Saliency Sandbox provides various types of features and activation maps. The color features are based on the actual color value of a pixel, while the orientation is extracted by applying Gabor filters. The motion class provides feature extractors based on sequential images. Activation maps extract the saliency in a single feature map. Moreover, the Saliency Sandbox provides some of the best-known approaches such as the GBVS, the center-surround differences provided by Itti & Koch, BMSs based on different levels of thresholding, and the FTSs or SWSs as frequency-based approaches out of the box. All activation maps can be applied to all features, leading to a tremendous number of possible combinations. Different kinds of activation maps can be combined into one saliency map.



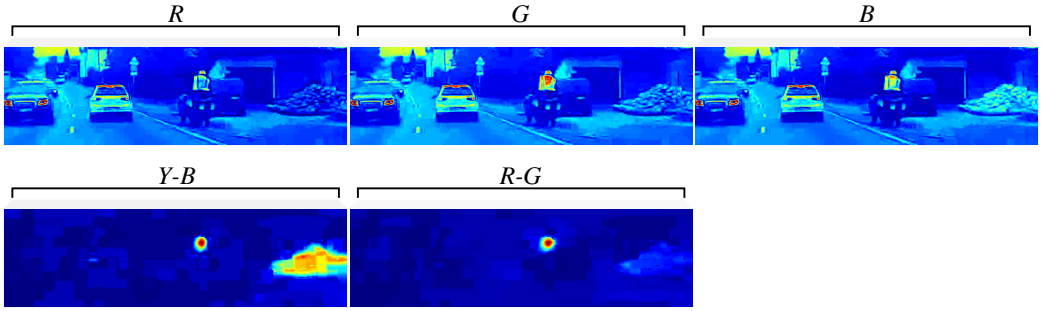
Fig. 5.1.4: Input image showing a driving scene. The motorcyclist with the yellow warning vest should have a high saliency as expected.

Intensity: The intensity is calculated according to the ITU BT.601 standard, i.e., as a weighted sum of the RGB channels [245]:

$$I = 0.299 \times R + 0.587 \times G + 0.114 \times B \tag{5.1}$$

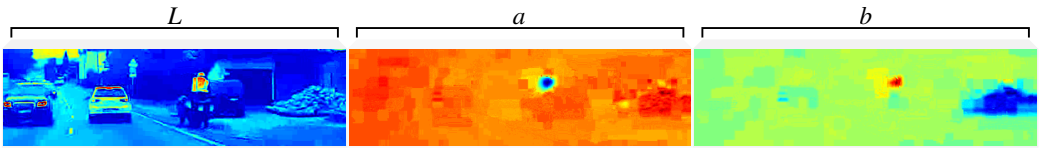


RGB: Features from the RGB color space can be the RGB channels themselves. However, inspired by the structure of the retinal ganglion cells, Itti & Koch extracted the difference of yellow[1]and blue ($Y - B$) as well as red and green ($R - G$) as additional features [243].

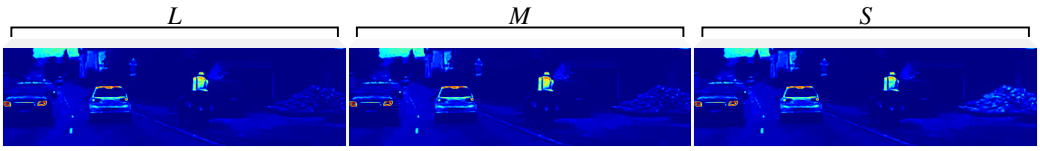


Lab: The Lab color space describes the luminance in the channel L, as well as the opponent colors red-green (channel a) and yellow-blue (channel b). Therefore the input image is transformed into the CIE-XYZ color space, yet using an equidistance in the change of color values and the empirical human color perception. Both, the FTS and BMS use the LAB color space as features [242], [244].

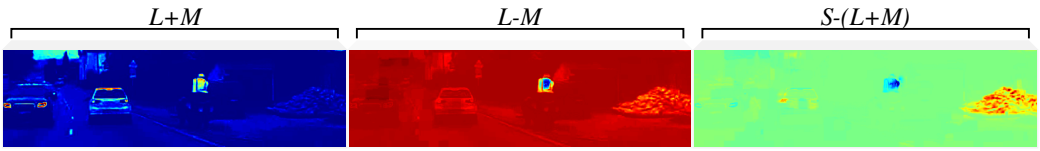
[1] $Y = R + G$



LMS: The LMS color space describes the response of the three cones of the human eye. The L-channel represents the *long-wavelength receptor*, which is responsible mainly for the red light perception. M represents the *medium-wavelength receptor*, which covers the light between blue and orange. The last channel S represents the *short-wavelength receptor* and covers blue light.



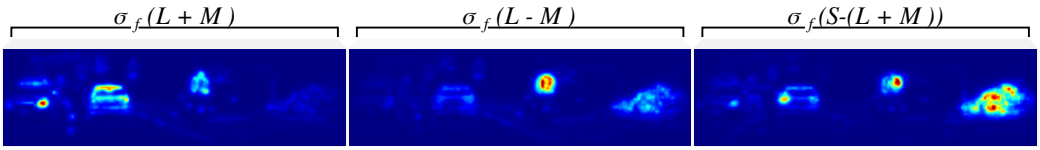
DKL: The Derrington-Krauskopf-Lennie color space is a cartesian representation of the LMS color space. The DKL color space respects the linear combination model of signals from the cones to perceive colors and luminance. Along the $I + M$ axis, only the perceived luminance changes, without affecting the chromaticity. The axis $I - M$ varies the chromaticity without changing the excitation of the blue-sensitive cone. Along with $S - (I + M)$ the chromaticity varies without impact to the red-green sensitivity [246].



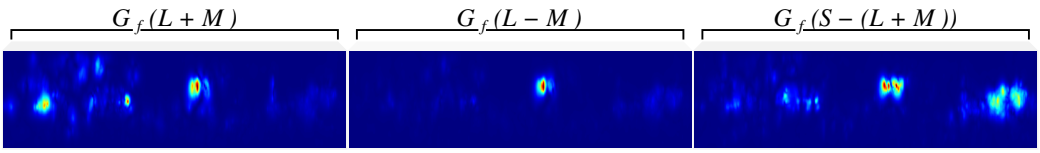
Contrast: The contrast feature $\sigma_f(I)$ is calculated as a spatial standard deviation. Therefore, a Gaussian filter estimates the local mean $\hat{E}(I)$. The local standard deviation is calculated as

$$\sigma_f(I) = \sqrt{\hat{E}\left(\left(I - \hat{E}(I)\right)^2\right)}. \tag{5.2}$$

The variance σ^2 of the Gaussian filter determines the locality of the standard deviation.

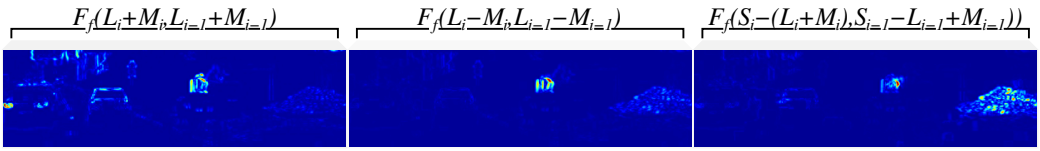


Orientation: Many saliency maps extract orientational features since Treisman and Glades discovered that the orientation of objects is related to the visual stimuli [247]. Hence, Itti & Koch, as well as J. Harel (GBVS), use Garbor filters $G_f(I)$ parametrized to different rotations to encode local orientation contrast [240], [243].

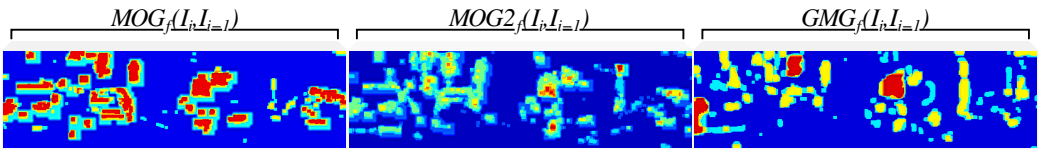


Flicker: The method by Harel et al. [248] provides flicker as an optional feature w.r.t. motion in sequential images. The Flicker feature is defined as the absolute difference of two subsequent images [248]:

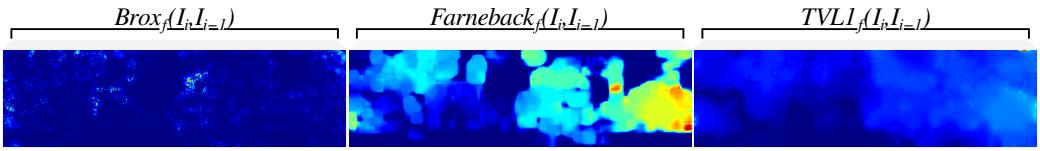
$$F_f(I_i, I_{i-1}) = \sqrt{(I_i - I_{i-1})^2}. \tag{5.3}$$



Background: Following the Flicker feature, the Saliency Sandbox provides various methods for background subtraction to extract motion features from sequential images. MOG_f and $MOG2_f$ employ Gaussian mixture models to describe the distribution of a background pixel over a sequence of images. The probable values of a pixel are those more static over the sequence [249], [250]. The GMG_f background subtraction uses a Bayesian model to detect foreground objects [251].



Optical Flow: As a further motion feature, the Saliency Sandbox provides several optical flow algorithms. Moreover, the optical flow is may high pass filtered to reduce the influence of ego-motion, especially in driving scenes. Therefore, a polynomial regression estimates the ego-motion, whereby recent images in the sequence are higher weighted than older ones.



In a typical setup, the extracted feature maps are forwarded as input to a succeeding activation model.

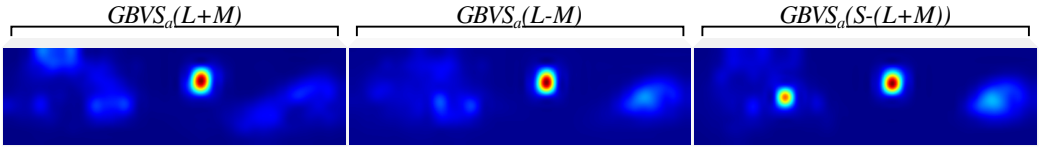
Activation Models

Activation models emphasize salient regions in the feature map. The Saliency Sandbox offers five different activation models, whereby each of them, bases on fundamentally different

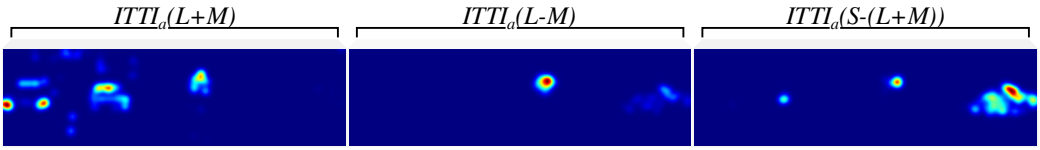
approaches.

Graph-based Visual Saliency (GBVS): The GBVS is inspired by the retinotopic network of the visual cortex. It creates a transition matrix from each pixel to all others. The transition weight is defined by the dissimilarity of two pixels, and the distance[2] of their spatial image location. The transition matrix is used as a Markov model, wherein the activation is calculated by iterative multiplication with an initial equally distributed vector (random walk) [240]. Considering the quadratic size of the transition matrix, the input feature usually needs to be massively downscaled. In the following example, the input was downscaled to 32 px × 32 px. However, in the context of dynamic saliency, the GBVS provides the advantage that the previous activation can be used as the initial solution of the new Markov system. This reduces the number of necessary iterations for each frame significantly.

The GBVS normalizes the activation by a concentration of mass. Therefore, a transition matrix is created again but with the activation and distance as transition weights. This normalization Markov system is again solved as above [240].



Itti & Koch: The approach by Itti & Koch et al. [243] uses center-surround differences to model the receptive field of a sensory neuron. The center-surround differences are implemented as a pyramid over different input scales. Subtracting the feature maps of two different scales results in a high response if a feature was noticeable on the larger scale, but not in the lower scaled feature map. Moreover, Itti & Koch suggest an iteratively applied $\mathcal{D}o\mathcal{G}$ filter followed by a rectification for normalization [243].

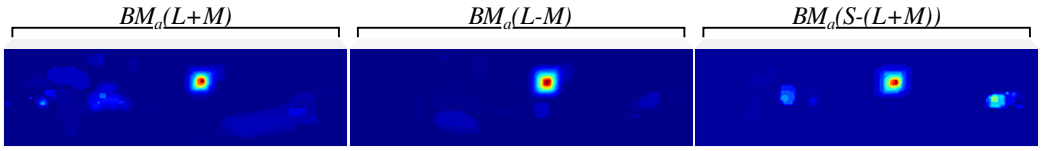


Boolean Maps Saliency (BMS): In contrast to the other activation maps, BMSs emphasizes salient features in the global image context. Therefore, several random applied thresholds binarize the input image into multiple feature maps. Thus, salient image regions are more often on the same threshold side, and thus more often share the same value in the binarized feature maps. The resulting binary maps are collected along with their complements. After applying morphological opening operations to remove noise and not enclosed objects, the binary maps are summed up to an activation map [244].

However, Saliency Sandbox's implementation does not use randomly generated thresholds, as this makes the result hard to compare and unstable in image sequences. Instead,

[2] The edge is weighted higher, the closer the distance between pixels.

the implemented version applies a fixed number of evenly spread thresholds between the minimum and maximum value in the feature map. An L^2-normalization realizes the normalization.

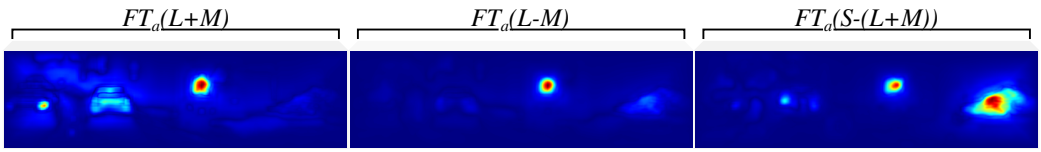


Frequency Tuned Saliency (FTS): The FTS calculates the Euclidean distance between the Lab color value of a Gaussian filtered input and the average Lab color value [242]. Since the average color value is calculated over the whole image, the FTS saliency is counted as a global saliency detection, similar to the BMS.

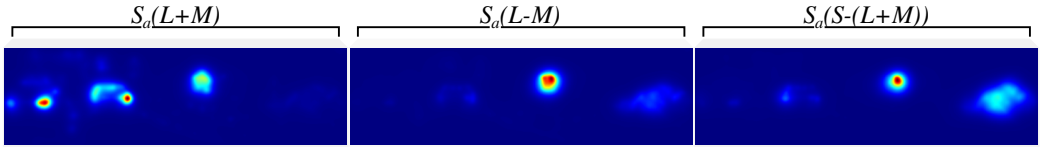
However, the Saliency Sandbox's implementation uses a heavily low-pass filtered input image, instead of the average color. Thus, the modified implementation also detects saliency in a local context of the image. Moreover, the calculation of the Euclidean distance over the channels is not implemented, as the framework processes the activation separately and independently on each channel.

However, setting $\sigma^2 = \infty$ for the second low pass filter, and calculating the activation $FT_a(I)$ over each channel in the Lab color space, the implementation fully complies with the original by using the following composition:

$$\sqrt{FT_a(L)^2 + FT_a(a)^2 + FT_a(b)^2} \tag{5.4}$$



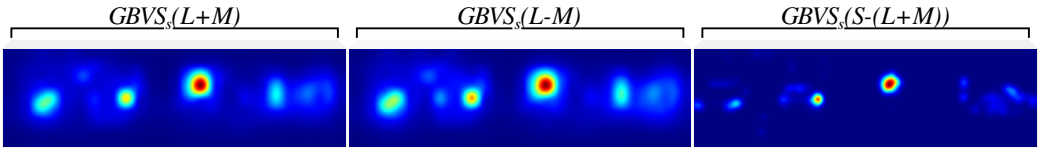
Spectral Whitening Saliency (SWS): This frequency-based approach from Hout et al. was reimplemented based on the *Spectral Saliency Toolbox* from Schauerte et al. [241], [252]. It analyzes the log spectrum of the input image to obtain the spectral residual and transform it into a spatial domain to predict the activation.



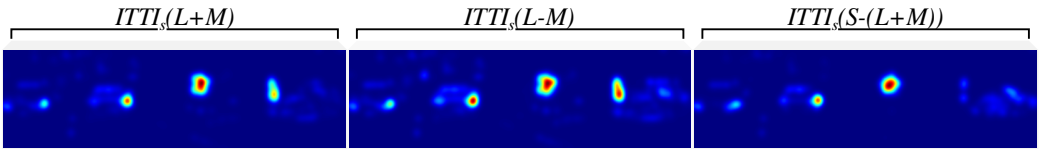
Composite

A common approach is to combine multiple activations based on distinct feature maps into a single output saliency map. All state-of-the-art saliency models provided by the Saliency Sandbox use weighted sums for the final activation map combination.

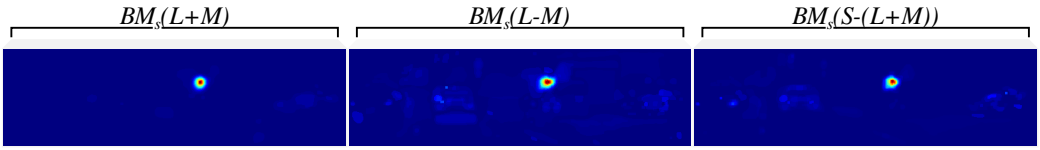
Graph-based Visual Saliency (GBVS): The GBVS combines color and intensity features as luminance, red-green, and blue-yellow along with orientation features. The activation is calculated in different scales and combined by the mean over all of them. As a default, the implementation from J. Harel et al. uses the DKL colorspace, corresponding to the last example below [248].



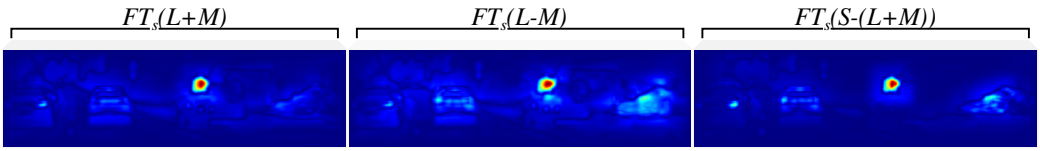
Itti & Koch: Itti & Koch use seven activation maps. The first three calculate the center-surround differences in the red-green, blue-yellow, and intensity feature map. As the GBVS, the Itti & Koch map uses four orientation features (0°, 45°, 90°, 135°) [243].



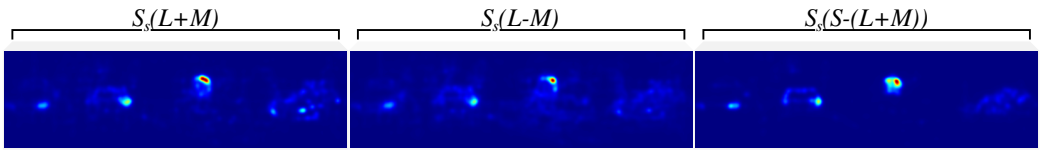
Boolean Maps Saliency (BMS): Zhang et al. use the three Lab color channels as features and combine them by calculating the mean over the three activation maps [244].



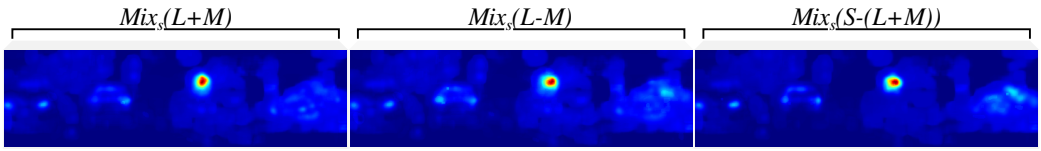
Frequency Tuned Saliency (FTS): The FTS was designed as a bandpass filter in the Lab color space [242]. As previously mentioned, the original implementation uses the Euclidean distance to composite the activation map of each color channel to one saliency map. The examples below show the result by just summing the activation maps.



Spectral Whitening Saliency (SWS): Hou et al. do not suggest specific feature maps in their work [241]. However, the implementation in the *Spectral Saliency Toolbox* provided by Schauerte et al. [252] calculates the mean of the normalized activation map over each input image channel. The examples below show the result of the SWS using different color spaces.



Mixed The Mixed saliency model demonstrates the flexibility of the Saliency Sandbox. It combines activations based on SWS, Itti & Koch, BMS, and Farneback motion.



5.1.3 Usage

One of the Saliency Sandbox's significant goals is to speed up research by providing a framework that covers a considerable set of features, activations, and utilities. Furthermore, the framework is fast and easily extendable, enabling efficient prototyping and implementation of new approaches. The following provides three code snippets that demonstrating how the Saliency Sandbox can be extended and applied to functionalities.

Define new activation: The following snippet demonstrates the definition of a new activation map. The feature map type is passed as a template argument enabling a fixed memory layout for embedded applications. The base class *Saliency* derives from the class *Algorithm*, as introduced in section 5.1.1. Default parameters are defined in the *reset* function, which is called by the abstract base whenever the algorithm should be reset or initialized. The actual calculation of the activation takes place in the *calc* function.

```
template<uint16_t _width,              // input/output width
         uint16_t _height,             // input/output height
         template<uint16_t,uint16_t> //
            typename _feature>         // feature type
class MyActivationSaliency: public Saliency<_width, _height> {
    private:
    _feature<_width,_height> m_feature;
    public:
    void reset() {
        this->setString("text",""); // set default
    }
    void calc() {
        // do something with the property
        std::cout << this->getString("text") << std::endl;
        // calculate the feature map
        this->m_feature.process(this->input());
        // calculate the output
        // calculate the activation map
        this->output(activate(this->m_feature.output()));
    }
```

```
};

```

Combine activation maps: The following snippet declares the class *MySaliency* deriving from *SumSaliency*, to combine multiple activation maps into a saliency map. The activation maps algorithm types are passed as template arguments and will be invoked by the *SumSaliency* in parallel using a thread pool. The individual activation results sum up in the output buffer. *SumSaliency* derives from the *Algorithm* base class and is therefore also usable in Matlab and Simulink. The abstract function *setParams* is called in the reset function and is supposed to configure the individual activation maps.

```
template<uint16_t _width,  // input/output width
         uint16_t _height> // input/output height
class MySaliency: public SumSaliency<_width,_height,
    // Itti & Koch activation on a DKL color feature
    IttiSaliency<_width,_height,DKLColorSaliency>,
    // New defined activation
    MyActivationSaliency<_width,_height,DKLColorSaliency>
    > {
    public:
    void setParams(uint16_t index, sal_t* sal) {
        switch(index) {
            case 0: // use the first dkl channel (L-M) as feature
                    // for the Itti & Koch activation
                sal->setInt("dkl.channel", 1);
                break;
            case 1: // MyActivationSaliency should say hello
                sal->setString("text", "hello world");
                break;
        }
    }
};

```

Create a tool: After creating a custom activation map and a new composition based on Itti & Koch activations, this last example shows how an executable may is created to examine a video saliency.

```
int main(int argc, const char* argv[]) {
    MySaliency<RES_HD> sal;

    // print help
    for(int i = 1; i < argc; i++)
        if(!strcmp(argv[i],"--help") || !strcmp(argv[i],"-h"))
            sal.printParams();

    algorithm_assert(argc >= 3);

    // parse algorithm parameters
    sal.parseParams(argc-3,&argv[3]);
```

```
13
14      // process video
15      sal.processVideo(argv[1],argv[2]);
16
17      return 0;
18  }
19
```

5.1.4 Limitations

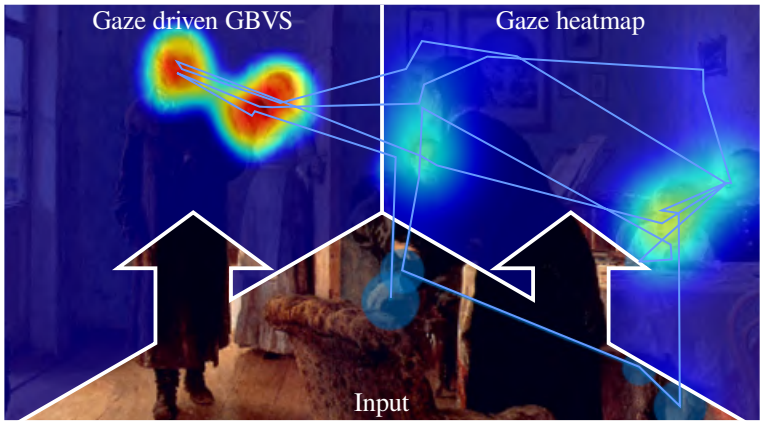
The Saliency Sandbox focuses on the development of static and dynamic bottom-up saliency maps. Yet, top-down approaches may use the Saliency Sandbox's algorithms to incorporate visual saliency and scene dynamics into their semantic interpretation, e.g., as introduced in Section 2.4.1. Moreover, due to the flexible architecture, a composition node may use the propagated feature activations to perform, e.g., a Neural Network (NN) for a non-linear combined saliency map. Such a complex classifier may include non-visual features like application domain, task, and user preferences (a similar approach is implemented in Section 6.2).

However, in the context of predicting an individual user's visual perception, the major limitation is the strong generalization of bottom-up salience maps. Traditionally, saliency maps predict the users' average foveal attention of a static or dynamic stimulus. Yet, to predict the foveal perception of a specific user, it becomes crucial to incorporate the user's line of sight. Moreover, state-of-the-art saliency maps' bio-physiological accuracy is limited, as human's retinal perception is a continuous process of visual stimulus processing and shifting attention. Hence, the next Section 5.2 introduces a modified version of the GBVS, incorporating a user's line of sight and continuous processing of the visual stimulus in an online fashion.

5.1.5 Conclusion

This section introduced the Saliency Sandbox, a flexible and performant C++ framework for runtime efficient saliency processing. Interfaces to Matlab and Simulink addresses a broad audience from scientific research up to deployment and prototyping in realtime signal processing environments. Moreover, by its tree-like processing architecture, interfaces to Cuda, and an integrated thread pool, the framework enables massively parallelization. A standardized, yet not rigidly prescribed processing structure from feature extraction, activation processing, and composition, enables researchers and developers to implement novel approaches efficiently, by combining and mixing them with state-of-the-art approaches.

5.2 Gaze-driven GBVS saliency map



Traditional bottom-up saliency maps predict the average users' foveal perception based on intrinsic scene features, e.g., by a retinotopic replication of the foveal stimulus modulation, e.g., by using the previously introduced Saliency Sandbox in Section 5.1. Although state-of-the-art approaches may incorporate scene dynamics, they do not reflect individual and temporal user dynamics, e.g., the visual exploration – the chronological succession of the user's line of sight shift. When viewing a scene, humans frequently focus their attention, known as a fixation, before shifting to another area with a rapid eye movement known as a saccade. The selectivity of the focused scene locations is a highly optimized and developed process, mainly driven by two factors: 1st) visual scene features, extracted by the retina (bottom-up), and 2nd) the interpretation of the extracted features regarding their semantic value by higher cognitive processes, and subsequent identification of the next fixation target (top-down) [110], [243], [253]. Modeling and understanding this reciprocal process is a long-term core topic in cognitive psychology and the computer vision community [254].

While saliency maps, such as GBVS, successfully replicate substantive parts of the retinal feature extraction and modulation, the selectivity of visual attention is directly measurable using eye-tracking. Saliency maps are predominantly bio-physiologically inspired algorithms to predict potential fixation targets [162]. Therefore, regions emitting a strong, recognizable visual stimulus are identified and emphasized by replicating the retinal visual stimulus processing. Eye-tracking, on the other hand, enables measuring the scene point of the user's visual attention directly, as introduced in Section 2.1. A widely used method to analyze the user's eye-tracking signal is Gaussian heatmaps. Therefore, each measured scene fixation is processed by a Gaussian distribution, modeling the eye-tracker noise and the user's spatial foveal perception. Moreover, depending on the implementation, the modeled

fixation distribution may include fixation time or other meta-information as additional factors. Yet, fixation heatmaps are subject to some limitations. For instance, slight shifts in the eye-tracking signal make it challenging to identify the scene parts that attracted the visual attention and which information of the scene was actually perceptible. In addition, depending on the implementation, long or frequent fixations on the same scene region may lead to a high density in the fixation heatmap. Hence, it is assumed that these regions are particularly relevant to the spectator since a comparatively high amount of visual information was extracted. However, frequent or long fixations may also be caused by severe scene conditions such as low contrasts. Thus, the visual information may be harder to extract and therefore requires longer or more frequent fixations to be perceived.

The following sections are supported by a prior publication from Geisler et al. at the conference on Eye-tracking Research & Applications (ETRA) in 2020 [2]. Section 5.2.1 introduces a joint method of saliency and fixation analysis, by incorporating detected fixations from an arbitrary eye-tracking signal into the calculation of the GBVS attention map. The evaluation in Section 5.2.2 demonstrates that the resulting heatmap equilibrates the measured visual attention to the retinal-perceivable stimulus, and thus models the density of perceived information in the scene more accurately as pure fixation or saliency maps. The final Sections 5.2.3 and 5.2.4 state the limitations and conclusion.

5.2.1 Method

Similar to most saliency map approaches, the GBVS algorithm is divided into three consecutive steps [240]:

1. Extraction of a feature map M_t on a given image I_t.
2. Calculation of an activation map A_t based on M_t.
3. Normalization and combination of the activation map A_t.

The proposed approach amends step 2 by injecting the gaze signal g_t into the calculation of the activation map A_t. Steps 1 and 3 remain unchanged to the GBVS publication [240] and not further discussed here (see Section 5.1 for a more detailed elaboration). The subscript t indicates the time domain since the gaze signal is given as a time series of consecutive fixation points. However, it also simplifies the handling with dynamic stimuli, such as videos. The following assumes that for each time t, a corresponding gaze signal g_t exists, as well as a stimulus I_t, respectively, a feature map M_t.

The GBVS interprets the activation map as a state vector of a Markov model. The transition between two states is defined by a dissimilarity score over the feature map M_t. Thus, a random walk over the Markov model empowers those states that are dissimilar in the respective feature map. Analogous to the original GBVS, the dissimilarity between the two states i and j in the feature map M_t is defined as follow:

$$d_t\left(i,j\right) = \left|\log \frac{M_t\left(i\right)}{M_t\left(j\right)}\right|, \tag{5.5}$$

where $M_t(i)$ is the i-th value of the corresponding feature map M_t. The transition weight $w_t(i,j)$ between the two states i and j is defined as the product of their dissimilarity score $d_t(i,j)$ and a distance weight $F_w(i,j)$:

$$w_t(i,j) = d_t(i,j) \cdot F(i,j). \tag{5.6}$$

The distance weight adds a local sensitivity to the dissimilarity score. Thus, states that are dissimilar to their immediate neighborhood are emphasized while the impact of the dissimilarity score is attenuated with increasing distance. $F(i,j)$ is defined as an exponentially weighted square distance between the states i and j in their spatial dimension in the input image I_t:

$$F(i,j) = \exp\left(-\frac{(x(i)-x(j))^2 + (y(i)-y(j))^2}{2 \cdot \sigma}\right), \tag{5.7}$$

where $x(i)$ and $y(i)$ is the x- and y-coordinate of the i-th state in the respective input image I_t. The free parameter σ controls the shape of the exponential distance weight. The larger σ is chosen, the more weight is given to the dissimilarities of more remote states.

The final Markov transition matrix T_t is then assembled as follows:

$$T_t = \begin{pmatrix} 1 & w_t(0,1) & \dots & w_t(0,n) \\ w_t(1,0) & 1 & \ddots & w_t(1,n) \\ \vdots & \ddots & \ddots & \vdots \\ w_t(n,0) & w_t(n,1) & \dots & 1 \end{pmatrix}, \tag{5.8}$$

where n is the number of elements in the feature map M_t respective the input image I_t.

The activation map A_t is then calculated by k repeated multiplication with the transition matrix T_t:

$$A_t^{(k)} = T_t \cdot A_t^{(k-1)}. \tag{5.9}$$

5.2.1.1 Incorporate Gaze

Up to this step, the procedure follows the original GBVS algorithm. However, instead of initializing $A_t^{(0)}$ equally distributed, the gaze position is encoded as initial activation map:

$$A_t^{(0)} = q \cdot A_{t-1}^{(k)} + (1-q) \cdot (F(0,g_t), \dots, F(n,g_t)), \tag{5.10}$$

where $F(i,g_t)$ is the exponential weighted square distance between the recorded gaze position g_t and the spatial location of the i-th element in the activation map. In other words, the activation map is initialized by the measured visual activation from the eye-tracking signal. Additionally, parameter $q \in [0,1]$ controls the influence of the previously calculated activation map A_{t-1} into the initialization of $A_t^{(0)}$. Thus, for $q > 0$, $A_t^{(0)}$ encodes the recently measured visual attention, but also the history of previously predicted attention areas. This smooths the resulting activation map $A_t^{(k)}$ in the temporal domain and makes noise in the gaze signal less significant. However, it also poses a risk to generate a distorted activation

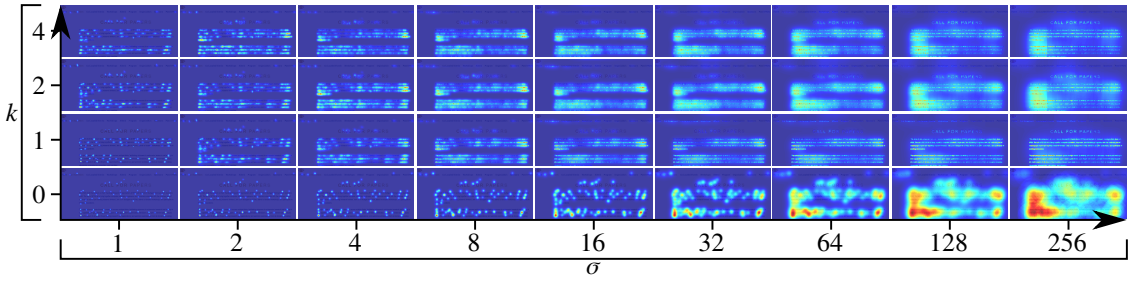


Fig. 5.2.1: Influence of the parameters σ (horizontal) and k (vertical) on the adaptation of the fixations heatmap to the text stimulus. The input is the same as used in Figure 5.2.2. For large k, the injected visual attention is increasingly distributed across the entire stimulus. The parameter σ should be chosen depending on the desired level of detail of the visualization. Using a text stimulus, it is usually reasonable to choose a high degree of detail (here $\sigma \leq 8$), in order to visualize the perception rate of single words or lines.

map. For instance, on a dynamic stimulus: the previous predicted attentive area in frame I_{t-1} is located somewhere in frame I_t. Yet, $A_t^{(0)}$ provides values in this area, and the Markov model will adapt it to the next salient region – which may not have been actually focused on. Nevertheless, this effect only occurs if the content of the scene changes significantly, for instance, at scene cuts in movies, or opening a new web page while browsing.

When generating static heat maps (such as Figure 5.2.3), it is common to ignore the temporal domain completely. In this case, q is set to zero. The overall heatmap $\mathbf{A}^{(k)}$ is then the weighted sum over $A_t^{(k)}$:

$$\mathbf{A}^{(k)} = \sum_t A_t^{(k)} \cdot b_t, \tag{5.11}$$

where the weighting b_t, for instance, can be chosen in relation to the fixation time.

5.2.1.2 Parameters

On regular gaussian like gaze heatmaps, σ models the area of visual attention (foveal perception) and/or the expected noise of the eye-tracking signal, and thus controls the acuity of the resulting heatmap. In the proposed approach, σ controls the distribution of visual attention deduced from the fixation signal. But also how far the Markov model may adopt this distribution to the underlying stimulus in each iteration. The number of iterations is controlled by the parameter k. Whereby for $k = 0$, $A_t^{(0)} = A_t^{(k)}$ corresponds to a regular gaussian like fixation heatmap of a single fixation point (respectively $\mathbf{A}^{(0)}$ overall fixation points). Figure 5.2.1 shows how the initial gaze heatmap $A_t^{(0)}$ is gradually distorted to the stimulus for each additional iteration over Equation 5.9.

5.2.1.3 Implementation Details

The main limitation of GBVS is runtime and memory consumption. The transition matrix T_t grows in quadratic size with the input size n, and thus quickly exceeds the available memory (e.g., $> 9.4 \cdot 10^{10}$ elements on a VGA resolution). Additionally, the initialization of T_t requires a runtime complexity of $\mathcal{O}(n^2)$. Both together, limit the GBVS to very low input resolutions, which leads to a loss of acuity. Thus, the standard parametrization of the GBVS toolbox limits the internal resolution to an edge length of 32 px [248].

On closer examination, however, it is apparent that most values in T_t are extremely small and have no significant impact to the resulting activation map $A_t^{(k)}$. Thus, after applying a threshold l, the transition matrix T_t becomes predominantly sparse. Furthermore, assuming that $M_t \in [0, 1]$, the elements of T_t, which potentially exceed the threshold l can be determined in relation to σ:

$$l < F(i, j), \tag{5.12}$$

and resolves to:

$$\sqrt{-2 \cdot \sigma \cdot \log(l)} \geq \sqrt{(x(i) - x(j))^2 + (y(i) - y(j))^2}, \tag{5.13}$$

where the right term is the euclidean distance between the i-th and j-th element in the feature map M_t. Thus, initializing T_t only requires the calculation of $2 \cdot \sqrt{-2 \cdot \sigma \cdot \log(l)}$ elements per row, since all other elements are not exceeding the threshold l. This reduces the actual runtime from $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$. Similar considerations can be made for the initialization of $A_t^{(0)}$ (although this is not a bottleneck). However, due to the sparseness of T_t and $A_t^{(0)}$, solving Equation 5.9 is much faster [255].

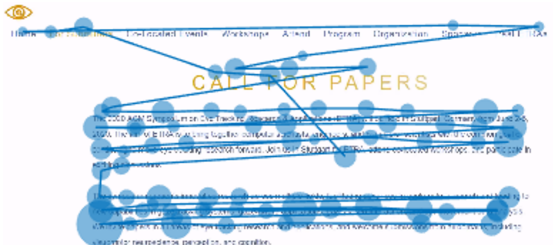
5.2.2 Evaluation

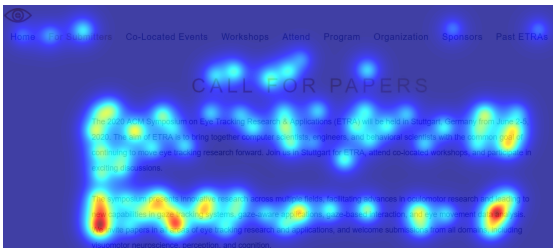
Figures 5.2.3, 5.2.2, and 5.2.4 demonstrate the application of the proposed visualization on different stimulus types: the *An Unexpected Visitor* painting from Ilya Repin, the *Call for Papers* website from ETRA 2020 as text, and a short video snippet of *Big Buck Bunny* from the Peach open movie project [256]. A Tobii Pro Spectrum recorded the gaze signal at 1200 Hz. The fixation locations and duration were extracted using the fixation filter I-VT provided by Tobii Pro Lab and default parametrization [257]. All stimuli were presented as full screen on the Monitor at 1920 px $\times$ 1080 px.

Considering the text stimulus, it is recognizable how the Markov model depicts the measured visual attention to paragraphs, lines, down to single words and characters. Therefore, the acuity of the heatmap is increased, and consequently, interpretations about the perception rate to text passages are simplified. For instance, in the field of web design and advertising, the proposed model can help to analyze whether a particular area attracts the desired level of visual attention and whether the presented information was quickly visual perceivable to the spectator.

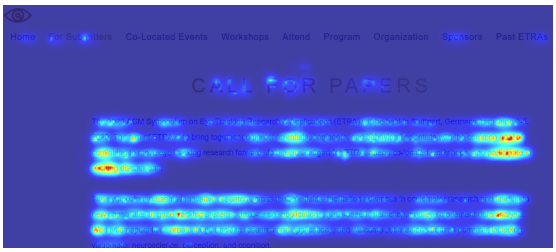
However, in this context, text reading is a relatively unambiguous challenge since the text is very salient to its background. At the same time, the text is often the only element that attracts the visual attention of the reader. The strength of the proposed method of

Input fixation map & stimuli





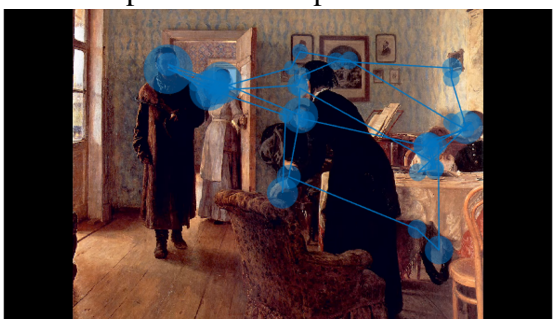
Gaze Heatmap ($\sigma = 32.0$)

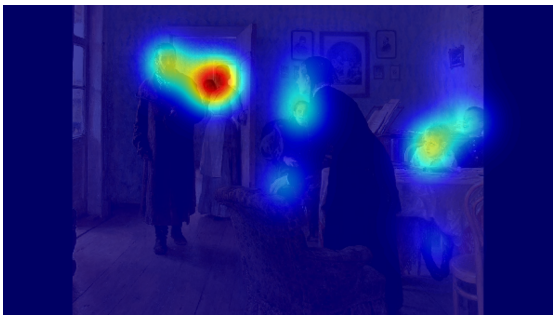


GBVS Heatmap ($\sigma = 4.0$, $k = 2$)

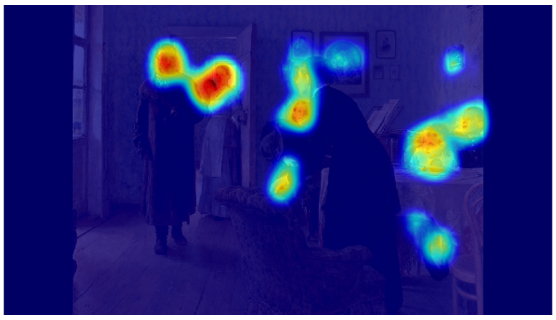
Fig. 5.2.2: The top image shows the recorded fixation sequence on the ETRA 2020's CFP website. The bottom left image shows a regular gaussian fixation heatmap ($\mathbf{A}^{(0)}$). The bottom right image shows the output of the proposed method ($\mathbf{A}^{(k=2)}$).

Input fixation map & stimuli





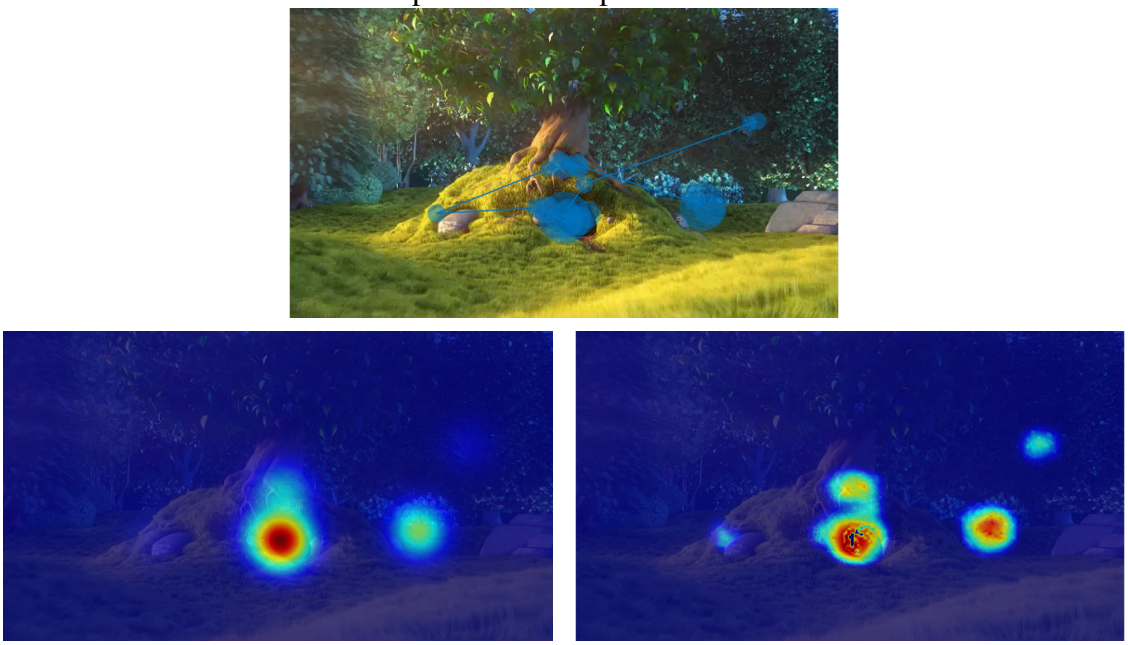
Gaze Heatmap ($\sigma = 32.0$)



GBVS Heatmap ($\sigma = 4.0$, $k = 2$)

Fig. 5.2.3: The top image shows the recorded fixation sequence on the painting *An Unexpected Visitor* from Ilya Repin (well known from the Yarbus experiment). The bottom left image shows a regular gaussian fixation heatmap ($\mathbf{A}^{(0)}$). The bottom right image shows the output of the proposed method ($\mathbf{A}^{(k=4)}$).

Input fixation map & stimuli



Gaze Heatmap ($\sigma = 32.0$) GBVS Heatmap ($\sigma = 4.0$, $k = 2$)

Fig. 5.2.4: The top image shows the recorded fixation sequence on a short snippet of the video clip *Big Buck Bunny* from the Peach open movie project [256]. The bottom left image shows a regular gaussian fixation heatmap ($\mathbf{A}^{(0)}$). The bottom right image shows the output of the proposed method ($\mathbf{A}^{(k=2)}$).

visual attention visualization is particularly evident in more complex stimuli, as shown in Figures 5.2.3 and 5.2.4. Considering the *An Unexpected Visitor* painting, the fixations are mainly on the faces in the scene, but also some miscellaneous areas, such as hands, the paintings in the background, or feet. However, the regular fixation heatmap has a particularly pronounced fixation cluster on the face of the woman in the background. This can be attributed to the fact that this face is particularly difficult to perceive due to its low contrast. Yet, the long and frequent fixations in this area lead to a suppression of all other fixations, which can lead to the interpretation that this area was of higher interest for the spectator. The GBVS generated fixation heatmap incorporates not only the fixation duration and frequency but also how accessible the stimulus in the region is to the observer. The result is a much more balanced fixation heatmap, where all the fixated heads are clearly pronounced.

5.2.3 Limitations

As with all bottom-up saliency maps, GBVS assesses the scene stimulus in terms of the foveal visual perception. Hence, the presented methodology is subjected to the same limitations. Although σ may control the spatially of the model, the distinct visual perceptual capabilities of different retinal regions are not depicted. Rather, a large σ implies the highly developed foveal perceptual abilities also apply to more peripheral regions. In the worst case, this leads

to false conclusions as to whether or not the user sufficiently perceived a specific scene area. At the same time, a too-small σ leads to an underestimation of the spatial extent of foveal perception.

The proper parameterization of sigma correlates to the user's distance to the presented scene stimulus. Hence, the following two Sections 6.1 and 6.2, present two approaches modeling the retinal perception capabilities beyond the mere foveal perception, incorporating the assessed stimulus' foveal eccentricity using a 3D representation.

5.2.4 Conclusion

The proposed method extends the well-known GBVS saliency algorithm by incorporating the measured visual attention. The resulting heatmap models a predicted perception rate of scene areas for an individual or multiple spectators. However, as the most bottom-up saliency algorithm, GBVS uses exclusively intrinsic scene features to predict whether certain scene content is attractive for fixation. It turns out that this is very accurate for a free viewing scenario. Yet, various tasks may require the spectator to direct their visual attention to less saliency scene areas. The proposed algorithm might distort these fixation points to a neighboring salient region and thus weigh the perception rate based on the wrong stimuli. This limitation can be compensated by using a small σ, and high-resolution scene images yet require high accuracy of the fixation point.

In practice, however, it has been shown that the proposed model generates more reasoned heatmaps than mere fixation heatmaps or saliency maps. Thus, the presented model provides an ingenious overview of the scene areas with a distinctive high rate of visual awareness.

The next Chapter 6 digs deeper into gaze driven perception models. While GBVS – gaze-driven or not – focuses on the replication of foveal perception, the models presented in the following chapter process the stimulus over the entire user's field of view by incorporating the line of sight and modeling distinct retinal capabilities.

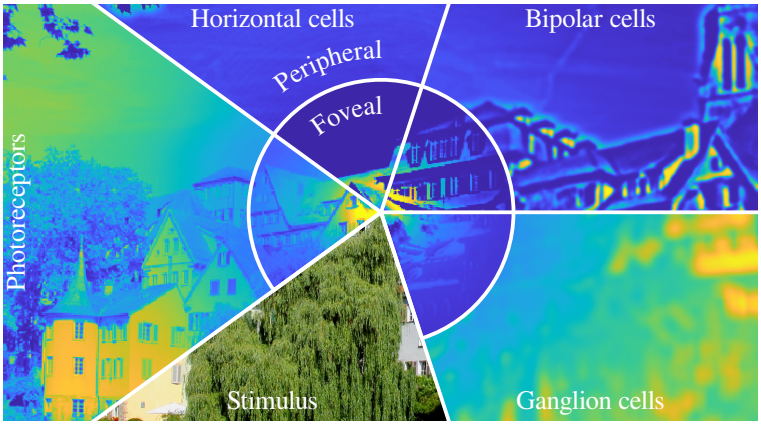
CHAPTER 6

Retinal model

The retinal feature extraction differs significantly according to the affected retinal area. The vast majority of chromatic photoreceptors are located in the fovea [258], [259], a comparatively small area ($\leq$ 1.5 mm$\varnothing$ vs. $\sim$32 mm$\varnothing$ of the total retina [260]–[262]), which provides the most detailed perception, and typically corresponds to the center of our visual attention in daylight conditions (photopic vision). With increasing eccentricity to the visual axis, the ability of chromatic perception decreases, and achromatic photoreceptors dominate perception. These photoreceptors react faster and more sensitively than chromatic photoreceptors, leading thus to higher sensitivity to movements and enabling perception even at low illuminance levels (scotopic vision). However, larger and more overlapping receptive fields come with increasing eccentricity, leading to a loss of spatial contrast in the peripheral field of view [121], [152], [263]–[267]. The subsequent underlying retinal neuronal network modulates the electrical stimulus arising from the photoreceptors over several layers and extracts the visual features of the scene. The last cell layer of the retina consists of ganglion cells that form the last stage of the receptive fields on the extracted stimulus and transmit these by the optic nerve to the visual cortex [268]. The reader is referred to Section 2.4 for a detailed break down of the neural stimulus modulation.

An accurate prediction of the perceivable scene content across a user's visual field requires meticulous replication of retinal perceptual capabilities. Moreover, such models are driven by a tracked line of sight to incorporate the scene's eccentricity towards the fovea, in order to parametrize the retinal stimulus processing. The following Section 6.1 presents a novel, highly accurate, bio-physiological inspired replication of the retinal stimulus processing. However, Section 6.2 introduces a lightweight derivate by replicating the retinal functional units to met the runtime requirements of challenging real-world tasks, such as driver monitoring.

6.1 Biophysiological model



The human visual perception of an optical stimulus results from the anatomical arrangement of the retinal photoreceptors and their interconnection up to the subsequent ganglion cells. Depending on the affected retinal area, the color respective wavelength and intensity of an optical stimulus evoke a different degree of neuronal activity and perception ability. This section proposes a biologically inspired model to predict the retinal visual perception transmitted to the higher cognitive processes via the optic nerve. Combined with an eye-tracking system and a scene camera, the proposed model enables a bio-physiologically reasoned and accurate prediction of the user's perceivability of scene objects. The model proceeds beyond conventional eye-tracking systems, which infer perception solely based on the explicitly fixated areas of a scene. Moreover, the presented model enables inference on perception beyond the line of sight, including the user's whole visual field of view. Thus, the model may determine whether a scene object is perceivable, even if it was not directly in the foveal area, i.e., explicitly fixated.

The following sections are supported by a prior publication from Geisler et al. at UNKNOWN in 2020 (UNDER REVIEW) **UNKNOWN** The methodology in Section 6.1.1 provides a detailed glimpse of the modeled neural processing of the visual stimulus in the spatial and temporal domain. Section 6.1.2 comprises a comprehensive evaluation, where an extensive driving simulator study provides the ground truth of perceived or not perceived objects. The evaluation reveals that the presented model enables the prediction of saccades, a couple of milliseconds in advance, by assessing the user's peripheral vision. Finally, the Sections 6.1.3 and 6.1.4 provide the limitations and the final remarks.

6.1.1 Method

The proposed model comprises two distinct sub-modules: 1) a probabilistic model of spatial retinal signal propagation (in Section 6.1.1.1), and 2) an accurate model of the retinal temporal signal modulation (in Section 6.1.1.2). Both modules may work independently, yet combined, they form a powerful and predictive spatio-temporal evaluation of retinal neuron activity.

6.1.1.1 A probabilistic model for retinal signal propagation

The retina provides more than 120×10^6 photoreceptors, converging to about 1.3×10^6 RGCs. The resolution of a RGC ranges from a single linked cone (MG) in the fovea up to $> 0.4\,\text{mm}$ diameter in the periphery. However, a complete simulation of all neural processes in the retina would exceed the computing capacities of most common systems. Moreover, this would require the visual stimulus in a sufficiently high resolution, which is seldom the case. Hence, the proposed probabilistic model computes the propagation of the visual stimulus from the photoreceptor cells through the different processing layers of the retina up to the ganglion cells and the optic nerve as a density function of neuronal response over a particular retinal area. In this context, the model examines a specific location θ of the retinal surface Θ, by estimating the likelihood of neuronal response based on the propagated visual stimulus' wavelength and intensity combined with the affected neurons' linkage and frequency. Therefore, the processing runtime is independent of the number of retinal neurons. Rather the runtime depends on the number of considered retinal locations to approximate the spatial density. Hence, the model is adaptable to runtime and acuity requirements. Moreover, the model enables to examine a particular area of the user's field of view by a partial retinal processing of the stimulus.

The signal flow of each retinal processing step is modeled as a conditional probability. $P(\theta|V)$ describes the probability that the neurons at the retinal coordinate $\theta \in \Theta$ emit a signal, given a propagated stimulus V. Using Bayes theorem, $P(\theta|V)$ can be computed as follows:

$$P(\theta|V) = \frac{P(\theta) \cdot P(V|\theta)}{\int_{\varphi\in\Theta} P(\varphi) \cdot P(V|\varphi)\ \mathrm{d}\varphi} \propto P(\theta) \cdot P(V|\theta)\,, \tag{6.1}$$

where $P(\theta)$ is the a-priori probability that the specific neurons are present at the retinal location θ. For many of the retinal neurons, physiological studies are available and report the frequency and linkage to other neurons (see Section 2.4.2). Depending on the type of the modeled neurons, the a-priori probability $P(\theta)$ is substituted with the normalized frequency $D(\theta)$ (e.g., Figure 2.4.7). The likelihood that a signal V is sufficient to activate the given neurons at the position θ, meaning signals are emitted to the subsequent layer, is denoted as the conditional probability $P(V|\theta)$. To make the formulas more readable, the constant normalization $\int_{\varphi\in\Theta} P(\varphi) \cdot P(V|\varphi)\ \mathrm{d}\varphi$ is not written explicitly in the following. The affected equations are indicated by the proportional character $\propto$.

The following describes a series of specializations of $P(\theta|V)$, which correspond to the transition probabilities of the signal V through the different processing layers of the retina: Photoreceptor cells (P) $\rightarrow$ HCs (H) $\rightarrow$ BCs (B) $\rightarrow$ RGCs (G); ACs are not considered

	Parameters	
	k_r	a_r (µm)
Rod	0.035 ± 0.007	3.142
L-Cone	0.027 ± 0.005	$1.767 - 28.274$
M-Cone	0.032 ± 0.007	$1.767 - 28.274$
S-Cone	0.037 ± 0.011	$1.767 - 28.274$

Tab. 6.1: Parameter overview to calculate the absorption rate in formula 6.2 and 6.3. k_r refers to the absorption coefficient at the maximum sensitivity of the photoreceptor [123]. a_r indicates the surface area of the photoreceptor. For cones, it varies from $\sim$1.767 µm in the foveal region (even smaller than rods) to $\sim$28.274 µm peripherally [269].

yet. Since, with a few exceptions, little is known about their quantitative influence and their actual function. Like HC, they interconnect BCs and RGCs laterally. Thus, the signal flow of the amacrine cells can be modeled analogously to the HC, as shown in the following, if required.

Photoreceptor cells absorb the incident light V_I and emit the signal V_R. The probability that a photoreceptor cell at the retinal position θ hyperpolarizes, and thus emits the signal V_R to the next processing layer, depends on the wavelength and amplitude of the incident light beam:

$$P_r\left(\theta|V_I\right) \propto \underbrace{a_r \cdot D_r\left(\theta\right)}_{P_r(\theta)} \cdot P_r\left(V_I|\theta\right), \tag{6.2}$$

where $r \in R = \{\mathrm{R}, \mathrm{L}, \mathrm{M}, \mathrm{S}\}$ indicates the considered photoreceptor type. $P_r\left(\theta\right)$ describes the chance of a photon encountering the respective photoreceptor, given by the size of the outer segment a_r and frequency $D_r\left(\theta\right)$ of the photoreceptor (see Figure 2.4.3, 2.4.5, and Table 6.1). Further, $P_r(V_I|\theta)$ provides the rate of the absorbed photons of stimulus V_I by the photoreceptor. According to Beer's law, the absorption rate can be formulated as:

$$P_r\left(V_I|\theta\right) = \mathrm{Tr}\left(V_I\right) \cdot \left(1 - 10^{-k_r \cdot \mathrm{A}_r(V_I)}\right), \tag{6.3}$$

where the first two terms $\mathrm{Tr}\left(V_I\right)$ describe the permeability of the lens, and the absorption rate of a photon regarding the sensitivity $\mathrm{A}_r\left(V_I\right)$ of the photoreceptors opsin, as given in Figure 2.4.4. The constant k_r is the absorption coefficient at the maximal sensitivity of the photoreceptor (see Table 6.1).

However, in practice, it is rarely the case that the stimulus is provided as the number and wavelength of incident photons. Rather, the stimulus is usually present as an RGB image. In this case, $P_r\left(V_I|\theta\right)$ roughly estimated by the transformation from the RGB color space to the LMS color space [270].

Figure 6.1.1 shows the calculated signal propagation likelihood $P_r\left(\theta|V_I\right)$ of the different photoreceptors on a natural stimulus. The foveal perception ($< 2°$) is mainly driven by L-

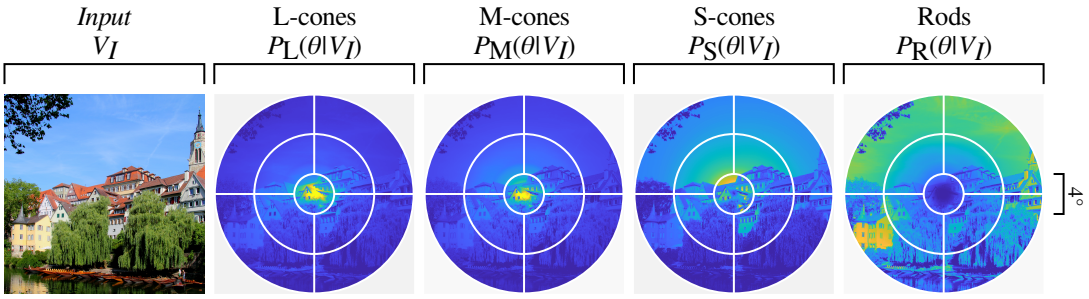


Fig. 6.1.1: Retinal signal propagation likelihood $P_r(\theta|V_I)$ of different photoreceptor types for an example stimulus V_I [271]. The retinal coordinate system range up to an eccentricity of 10° (2° inner circle, 6° middle circle), and was calculated with a resolution of 701 px × 701 px.

and M-cones. S-cones are predominantly active in the parafoveal area ($< 6°$), whereas rods increasingly dominate the peripheral vision.

Horizontal Cells (HCs) connect photoreceptors axons laterally in the OPL, meaning they may inhibit their hyperpolarization by increasing their membrane potential. $P_{\hat{r}}(\theta|V_R)$ describes the likelihood that a photoreceptor cell of type $\hat{r} \in R$ hyperpolarises, analog to $P_r(\theta|V_I)$, but includes the inhibitory effect of the HCs:

$$P_{\hat{r}}\left(\theta|V_R\right) \propto \underbrace{D_{\hat{r}}\left(\theta\right)}_{P_{\hat{r}}(\theta)} \cdot P_{\hat{r}}\left(V_R|\theta\right), \tag{6.4}$$

and

$$P_{\hat{r}}(V_R|\theta) = \sum_{h \in H} p_{\hat{r},h}\left(\theta\right) \cdot \mathrm{sig}\left(V_{\hat{r}}\left(\theta\right) - \mu_h\left(V_R, \theta\right)\right), \tag{6.5}$$

where $h \in H = \{\text{HC-A}, \text{HC-B}\}$ indicates the HC type. $p_{\hat{r},h}\left(\theta\right)$ describes the linkage ratio of the photoreceptor type $\hat{r}$ to the HC type h. To the best of the author's knowledge, there are no quantitative statements regarding the linkage frequency of HC-A and HC-B as a function of eccentricity. However, since HC-A are exclusively connected to cones and HC-B is predominantly connected to rods, proportional distribution of the HC types as the connected photoreceptors types (see Figures 2.4.3 and 2.4.5) is assumed. The chance of the raised photoreceptor potential V_R to exceed the aggregated potential $\mu_h\left(V_R, \theta\right)$ of the HC is modeled as a sigmoid activation function. The HC potential $\mu_h\left(V_R, \theta\right)$ is in turn, estimated by the weighted sum of the linked photoreceptor potential:

$$\mu_h\left(V_R, \theta\right) = \sum_{r \in R} p_{h,r}\left(\theta\right) \cdot \int_{\varphi \in \Theta} V_r(\varphi) \cdot \mathcal{N}(\varphi, \theta, \sigma_h\left(\theta\right)) \, \mathrm{d}\varphi, \tag{6.6}$$

where $p_{h,r}\left(\theta\right)$ is analog to $p_{\hat{r},h}\left(\theta\right)$, the linkage ration of the HC type h to the photoreceptor type r. The scaling factor $\sigma_h\left(\theta\right)$ indicates the dendritic tree size of the respective HC type at

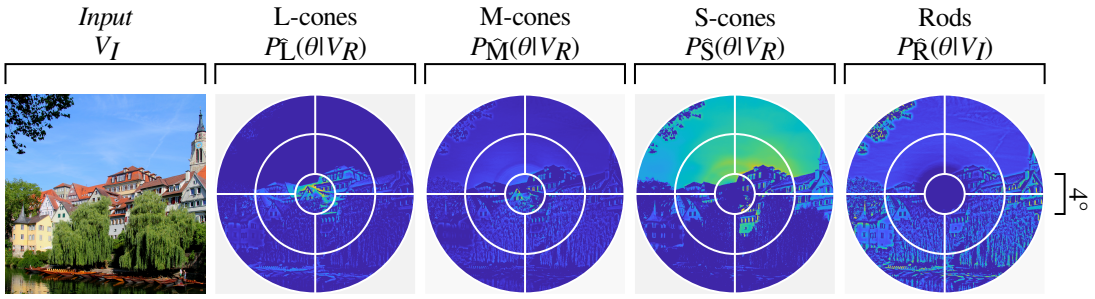


Fig. 6.1.2: Retinal signal propagation likelihood $P_{\hat{r}}(\theta|V_I)$ of different photoreceptor types, including the HCs for an example stimulus V_I [271]. The retinal coordinate system range up to an eccentricity of 10° (2° inner circle, 6° middle circle), and was calculated with a resolution of 701 px × 701 px.

θ. Depending on the HC type, $\sigma_h(\theta)$ is chosen between 0.77 mm and 1.04 mm for HC-A, or between 3.64 mm and 6.59 mm for HC-B, in relation to the retinal eccentricity $|\theta|$.

Figure 6.1.2 shows the calculated signal propagation $P_{\hat{r}}(\theta|V_I)$ of the different photoreceptors with respect to the linked horizontal cells. Horizontal cells form the first stage of the retinal receptive fields leading to an increase in spatial contrast.

Bipolar Cells (BCs) receive the signal from the photoreceptors and HCs ($\hat{V}_R$), and transmit it as ON or OFF to the subsequent ganglion cells. The probability that a BC at location θ emits a signal V_B, given the signal from the photoreceptors and HCs $\hat{V}_R$, is defined as following:

$$P_b\left(\theta|\hat{V}_R\right) \propto \underbrace{D_b(\theta)}_{P_b(\theta)} \cdot P_b\left(\hat{V}_R|\theta\right), \tag{6.7}$$

where b indicates the type of the BC. Along with the cell types listed in the Table 2.1, the FMB and IMB are further subdivided into FMB-L, IMB-M, FMB-S, IMB-L, and IMB-M, as they selectively connect different cone types: $b \in B = \{\text{FMB-L}, \text{FMB-M}, \text{FMB-S}, \text{IMB-L}, \text{IMB-M}, \text{DB1}, \ldots, \text{DB6}, \text{RB}, \text{BB}\}$. The frequency $D_b(\theta)$ of a bipolar cell type b at location θ is estimated from the frequency of the linked photoreceptors (see Figure 2.4.3, 2.4.5, and Table 2.1) and RGCs (see Figure 2.4.6, and Table 2.2). $P_b\left(\hat{V}_R|\theta\right)$ provides the likelihood of the activation of the bipolar cell as a sigmoid function:

$$P_b\left(\hat{V}_R|\theta\right) = \sum_{r\in R} p_{r,b}(\theta) \cdot \begin{cases} \text{sig}\left(\mu_b\left(\hat{V}_R,\theta\right) - \tau_b\right) & \text{if } b \text{ is ON} \\ 1 - \text{sig}\left(\mu_b\left(\hat{V}_R,\theta\right) - \tau_b\right) & \text{if } b \text{ is OFF} \end{cases}, \tag{6.8}$$

where $p_{r,b}$ is the linkage ratio between the BC type b and photoreceptor type r as shown in Table 2.1. As with HCs, the activation potential is modeled as the weighted sum over the

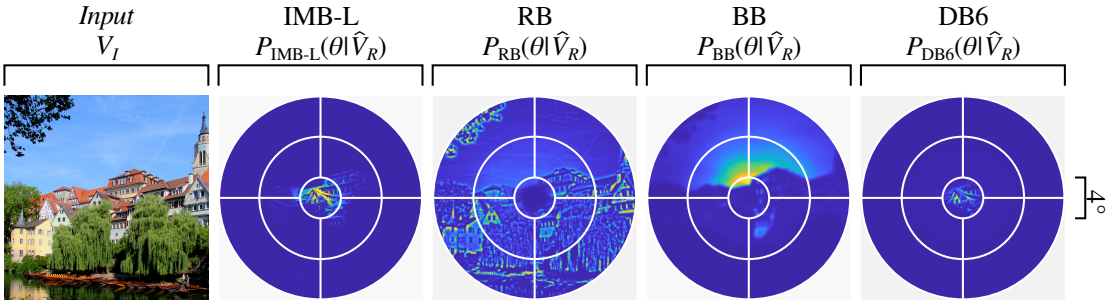


Fig. 6.1.3: Retinal signal propagation likelihood $P_b(\theta|V_I)$ of different BC types for an example stimulus V_I [271]. The retinal coordinate system range up to an eccentricity of 10° (2° inner circle, 6° middle circle), and was calculated with a resolution of 701 px × 701 px.

linked photoreceptors:

$$\mu_h\left(\hat{V}_R, \theta\right) = \sum_{r \in R} p_{b,r}\left(\theta\right) \cdot \int_{\varphi \in \Theta} \hat{V}_r\left(\varphi\right) \cdot \mathcal{N}(\varphi, \theta, \sigma_b\left(\theta\right)) \, \mathrm{d}\varphi, \tag{6.9}$$

where $p_{b,r}\left(\theta\right)$ is analog to $p_{r,b}\left(\theta\right)$, the linkage ration of the BC type b to the photoreceptor type r. The scaling factor $\sigma_b\left(\theta\right)$ indicates the dendritic tree size of the respective BC type at θ.

Figure 6.1.3 shows an example of the signal propagation likelihood of four different BCs.

Retinal Ganglion Cells (RGCs) emit a signal when the potential in the center of their receptive field exceeds the periphery. Depending on the RGC type, they can be found either with an ON center or OFF center. A widespread approach is to model the response intensity of the RGC as a convolution of a $\mathcal{DoG}$ on the incoming stimulus. This corresponds essentially to a subtraction of the center with its periphery (receptive field). This emphasizes local spatial intensity gradients, while large gradients are suppressed, which corresponds to the high-pass filtering of the input stimulus [110], [154]. However, technically the RGCs does not subtract the potential but sum the potential of ON and OFF BC channels. Thus, the aggregated potential of an ON center and an OFF periphery (or vice versa) is encoded and propagated to the optic nerve. Accordingly, the likelihood of signal propagation is modeled as the joint likelihood of its frequency and activation:

$$P_g\left(\theta|V_B\right) \propto \underbrace{D_g\left(\theta\right)}_{P_g(\theta)} \cdot P_g\left(V_B|\theta\right), \tag{6.10}$$

where $g \in G = \{\text{MG-L-OFF}, \ldots, \text{MG-S-ON}, \text{PG-ON}, \text{PG-OFF}, \text{BG}\}$ indicates the type of the RGC (see Table 2.2). Similar to the MB, the MG-OFF/ON are further split into MG-L-OFF, ... , MG-S-ON to incorporate the chromatic selectivity. The a-priori likelihood of a certain RGC type g at location θ is given by the frequency $D_g\left(\theta\right)$ as shown in Figure

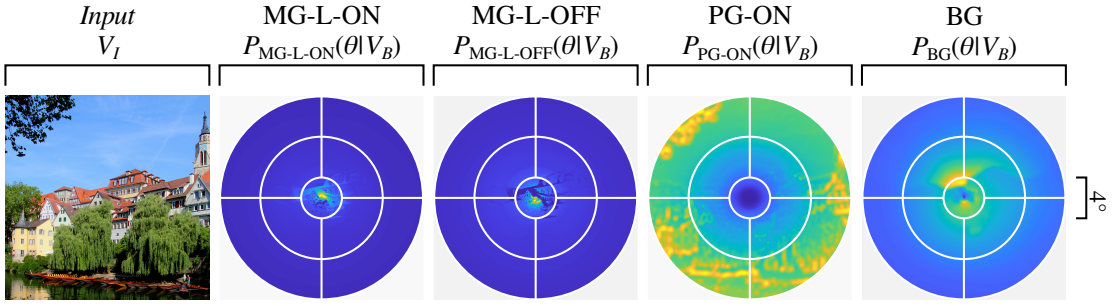


Fig. 6.1.4: Retinal signal propagation likelihood $P_b(\theta|V_I)$ of different RGC types for an example stimulus V_I [271]. The retinal coordinate system range up to an eccentricity of 10° (2° inner circle, 6° middle circle), and was calculated with a resolution of 701 px × 701 px.

2.4.6. The likelihood of the signal propagation $P_g\left(V_B|\theta\right)$ is modeled by a sigmoid activation function:

$$P_g\left(V_B|\theta\right) = \text{sig}\left(\mu_{g,\text{center}}\left(V_B,\theta\right) + \mu_{g,\text{periphery}}\left(V_B,\theta\right) - 1\right), \tag{6.11}$$

where $\mu_{g,\text{center}}$ and $\mu_{g,\text{periphery}}$ are the aggregated signals in the center and periphery respectively:

$$\mu_g\left(V_B,\theta\right) = \sum_{b\in B} p_{g,b}\left(\theta\right) \cdot \int_{\varphi\in\Theta} V_b\left(\varphi\right) \cdot \mathcal{N}(\varphi,\theta,\sigma_g\left(\theta\right))\,\mathrm{d}\varphi, \tag{6.12}$$

where $p_{g,b}\left(\theta\right)$ is the linkage ratio of the RGC type g to the BC type b either in its center ($\mu_{g,\text{center}}\left(V_B,\theta\right)$) or periphery ($\mu_{g,\text{periphery}}\left(V_B,\theta\right)$) as given in Table 2.2. The dendritic tree size is modeled by $\sigma_g\left(\theta\right)$ analog to Figure 2.4.7, where $\sigma_g\left(\theta\right)$ is larger in the periphery as in the center.

Figure 6.1.4 shows the propagation likelihood of two MGs, one PG, and the BG types. It is apparent that the MGs only have an increased rate of signal transduction in the foveal area. However, the signal propagation probability of BGs is predominantly in the parafoveal region, while PGs cover the outer foveal region.

6.1.1.2 Signal generation

The previous section presented a likelihood model to compute the spatial signal flow, including the frequency of the different neuron classes and their interconnections. The likelihood of retinal signal transduction is defined as the conditional probability $P\left(\theta|V\right)$ of a given stimuli V in the retinal region θ. The following section completes the model by including temporal effects such as the hyperpolarization of photoreceptors, signal impedance of the membrane, or the phasic or tonic response of BCs, encoded as a time-dependent signal $\mathrm{V}\left(\theta,t\right)$. The model, therefore, does not reflect the exact membrane potential of individual neurons, but rather temporal effects on the signal modulation. For this purpose, the conditional probability for a signal emission $P(V|\theta)$ is modulated in a temporal manner using differential equation systems, whose integration over the time t leads to the estimation of

the signal amplitude at a retinal area θ.

Photoreceptors represent the beginning of retinal signal transduction. Light-sensitive opsin molecules in the outer segment of the photoreceptor (rhodopsin in rods, L-, M- or S-iodopsins in cones) absorb incident photons and activate a cascade of chemical reactions which finally leads to a modulation of the membrane potential V_R in the photoreceptor.

The following equation system is adopted from the model from Hennig et al. from 2002 [163] to simulate the signal transduction inside the photoreceptor cell. The concentration of the activated opsin in a photoreceptor at θ is expressed as:

$$\tau \cdot \frac{\partial \mathrm{O}_r(\theta,t)}{\partial t} = P_r(\theta|V_I) \cdot (1 - \mathrm{O}_r(t)) - \mathrm{O}_r(t), \tag{6.13}$$

where $\mathrm{O}_r(\theta,t)$ is the concentration of activated opsin, τ is a time constant, and $P_r(V_I,\theta)$ describes the likelihood of an absorbed photon at location θ, thus the opsin triggers the following cascade and a signal will be emitted at θ. An activated opsin amplifies the signal by triggering $A_{\mathrm{T}_r} = \sim 10^3$ transducines:

$$\frac{\partial \mathrm{T}_r(\theta,t)}{\partial t} = A_{\mathrm{T}_r} \cdot \mathrm{O}_r(\theta,t) \cdot (1 - \mathrm{T}_r(\theta,t)) - \mathrm{T}_r(\theta,t), \tag{6.14}$$

where $\mathrm{T}_r(\theta,t)$ is the concentration of activated transducines. The activated transducines then leads to an increase in the PDE concentration:

$$\frac{\partial \mathrm{PDE}_r(\theta,t)}{\partial t} = \mathrm{T}_r(\theta,t) \cdot (1 - \mathrm{PDE}_r(\theta,t)) - \mathrm{PDE}_r(\theta,t), \tag{6.15}$$

where $\mathrm{PDE}_r(\theta,t)$ describes the concentration of the activated PDE. The PDE catalyzes the hydrolysis of the messenger cGMP by converting it to 5'GMP. A single PDE hydrolyses about $A_{\mathrm{cGMP}_r} = 10^3$ times as many cGMP molecules, which leads to a closure of the Ca^{2+} and Na^+ channel, which in turn influences the resynthesis of the cGMP concentration:

$$\begin{aligned}
\frac{\partial \mathrm{cGMP}_r(\theta,t)}{\partial t} =& A_{\mathrm{cGMP}_r} \cdot \mathrm{PDE}_r(\theta,t) \cdot (1 - \mathrm{cGMP}_r(\theta,t)) \\
& - \mathrm{Ca}_r^{2+}(\theta,t) \\
\frac{\partial \mathrm{Ca}_r^{2+}(\theta,t)}{\partial t} =& A_{\mathrm{Ca}_r^{2+}} \cdot \mathrm{cGMP}_r(\theta,t) \cdot (1 - \mathrm{Ca}_r^{2+}(\theta,t)) \\
& - \mathrm{Ca}_r^{2+}(\theta,t) \\
\frac{\partial \mathrm{Na}_r^{+}(\theta,t)}{\partial t} =& A_{\mathrm{Na}_r^{+}} \cdot \mathrm{cGMP}_r(\theta,t) \cdot (1 - \mathrm{Na}_r^{+}(\theta,t)) \\
& - \mathrm{Na}_r^{+}(\theta,t)
\end{aligned}, \tag{6.16}$$

where $\mathrm{cGMP}_r(\theta,t)$ is the concentration of converted cGMP. The activation of the ion channels Ca^{2+} and Na^+ is inversely calculated by $1 - \mathrm{Ca}^{2+}(\theta,t)$ respective $1 - \mathrm{Na}^{+}(\theta,t)$. The membrane potential of the photoreceptor is nonlinear related to the activation of the Na^+

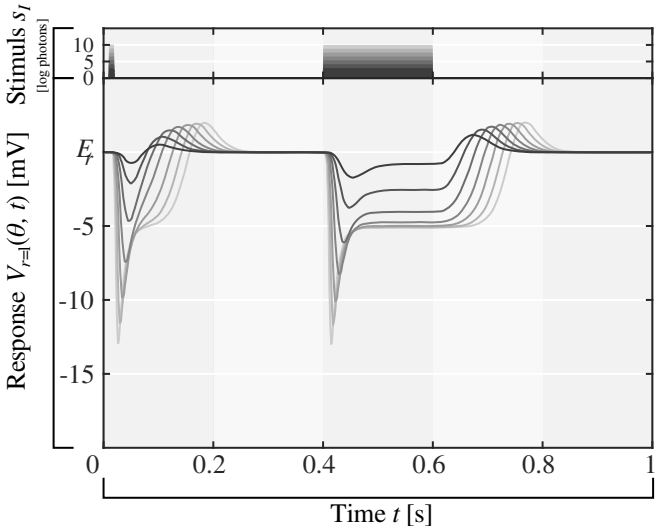


Fig. 6.1.5: Simulation of the L-Cone photoreceptor response at the retinal area θ using a variable number of photons and duration of irradiation. In the first part, a pulse of 10 ms, was simulated with 3 to 10 log photons at 560 nm wavelength. In the second part, an irradiation duration of 100 ms was simulated for the same number of photons.

and Ca^{2+} channels in the cell (K^+ flux was not considered here). Studies from [155]–[158] show a sudden increase in the membrane potential during hyperpolarisation of the cell, which then settles to a lower level. The membrane potential is, therefore modeled as an exponential increase of the membrane potential as followed:

$$\begin{aligned} \frac{\partial \mathrm{I}_r(\theta,t)}{\partial t} &= (10^{-\mathrm{Na}_r^+(\theta,t)} - 1) - \mathrm{I}_r(\theta,t) \\ C_m \cdot \frac{\partial \mathrm{V}_r(\theta,t)}{\partial t} &= q_{\mathrm{Na}_r^+} \cdot \frac{\partial \mathrm{Na}_r^+(\theta,t)}{\partial t} + q_{\mathrm{I}_r} \cdot \frac{\partial \mathrm{I}_r(\theta,t)}{\partial t} \end{aligned}, \tag{6.17}$$

where $\mathrm{I}_r(\theta,t)$ models the slope of the current flow while the hyperpolarization and $\mathrm{V}_r(\theta,t)$ gives the final membrane potential. Please note that in this model $\mathrm{V}_r(\theta,t)$ positively deflects in incident light and converges towards zero in darkness. A more realistic estimation of the membrane potential is obtained by the difference of the resting potential E_r and $\mathrm{V}_r(\theta,t)$:

$$\hat{\mathrm{V}}_r(\theta,t) = E_r - \mathrm{V}_r(\theta,t)\,. \tag{6.18}$$

Figure 6.1.5 shows the simulation of the photoreceptor response at different stimuli by solving equation (6.18). The peak that occurs during hyperpolarisation is well discernible, which then settles to a lower level. The number of photons influences both the reaction rate and the amplitude of the potential change, which coincides with the observations of Schneeweis and Schnapf in 1999 on macaques [156]. Figure 6.1.6 shows the simulated spatial-temporal signal propagation of L-cones and rods, including the hyperpolarization

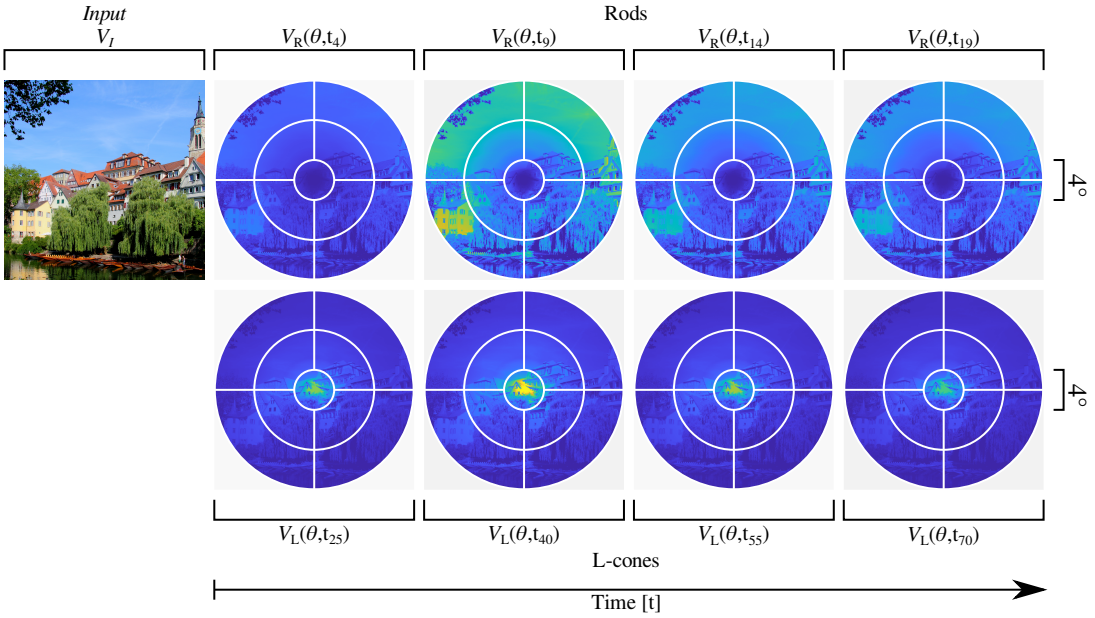


Fig. 6.1.6: Spatial-temporal signal generation. The second retinal image shows the simulated signal flow approaching hyperpolarization (please note the different scaling for L-cones and Rods). The last two images show the afterhyperpolarization period.

phase.

Horizontal Cells (HCs) modulate the signal by increasing or decreasing the potential of the photoreceptors' axon terminals by a horizontal linkage between multiple receptors. Thereby, they may inhibit the signal of a photoreceptor and thus adjust their sensitivity depending on their closer neighborhood. The likelihood $P_h\left(V_R|\theta\right)$ modulates this spatial influence to the signal density for a retinal area θ. Since HCs do not transmit the signal by itself to the bipolar cells, but merely have a passive effect on the signal generation of the photoreceptors, a feed-forward connection to bipolar cells is assumed for simplification:

$$C_m \cdot \frac{\partial \mathrm{V}_h\left(\theta,t\right)}{\partial t} = P_h\left(\theta|V_R\right) \cdot \left(1 - \mathrm{V}_h\left(\theta,t\right)\right) - \mathrm{V}_h\left(\theta,t\right), \tag{6.19}$$

where $h \in \{\text{L-A}, \ldots, \text{S-A}, \text{R-B}, \text{L-B}, \ldots, \text{S-B}\}$ describes the combination of the HC type and photoreceptor type, θ is the considered retinal area, V_R the signals from the different photoreceptors, and t the time of observation.

Figure 6.1.7 shows the simulated response of an L-HC-A on different exposures of peripheral and the center. The temporal pattern is consistent with the measurements of Dacey et al. in 2014 on macaques [134]. Figure 6.1.8 shows the spatial-temporal signal flow simulated on for L-cones (including L-HC-As and L-HC-Bs).

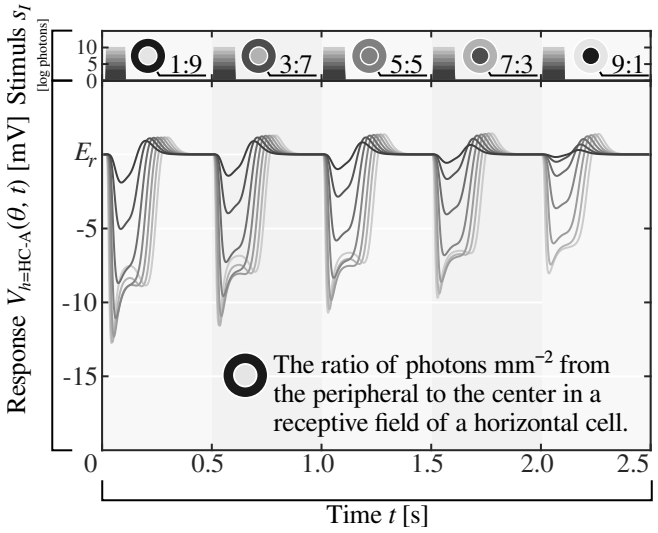


Fig. 6.1.7: Simulation of the potential change of the receptive field formed by the HC-As in the retinal area θ. The simulated stimulus consists of a pulse of 100 ms and 3 to 10 log photons (see Figure 6.1.5 for comparable photoreceptor response). The center consists of an L-cone photoreceptor, the periphery of S-, M-, and L-cones with a ratio of $6 : 47 : 47$. The proportion of photons in the center of the receptive field to the peripheral area decreases by each pulse from left to right. With an increasing number of photons in the peripheral area, it is more likely that the hyperpolarization of a photoreceptor in the center is inhibited, and the signal emission in the region θ is reduced.

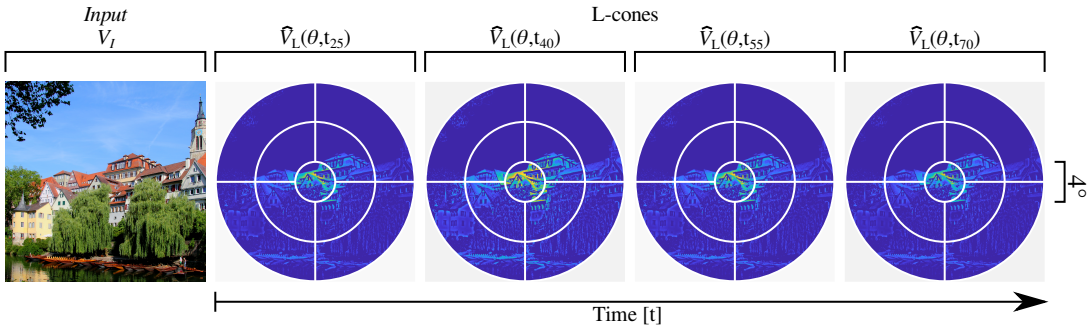


Fig. 6.1.8: Simulated spatial-temporal signal flow for L-cones (including L-HC-As and L-HC-B).

Bipolar Cells (BCs) differ according to their temporal behavior, whether the BC is sustained or transient. Sustained BCs emit a roughly constant potential for a constant stimulus. They, therefore, transmit the stimulus proportionally and adapt only slowly to variations; thus, they have a low-pass filtering effect to the input signal. Transient BCs, on the other hand, show a continuously reduced potential on a constant stimulus. They adapt quickly and reproduce the stimulus differential according to the speed of the stimulus change [159]. Is the intensity of the stimulus unchanged, the potential finally drops to the resting potential. Hence, transient BCs may also be interpreted as high-pass filters mechanism on the input signal:

$$\begin{aligned} \frac{\partial \mathrm{F}_b(\theta,t)}{\partial t} &= \left(P_b\left(\theta|V_H\right) - \mathrm{F}_b\left(t\right)\right) \cdot \tau_b^{-1} \\ \frac{\partial \mathrm{V}_b(\theta,t)}{\partial t} &= P_b\left(\theta|V_H\right) - \mathrm{F}_b\left(t\right) - \mathrm{V}_b\left(t\right), \\ \tau_b &= \begin{cases} \tau_b > 1 & \text{if } b \text{ is sustained} \\ \tau_b < 1 & \text{if } b \text{ is transient} \end{cases} \end{aligned} \tag{6.20}$$

where $P_b(\theta|V_H)$ describes the likelihood of an incoming stimulus from the HCs and photoreceptors and a therefrom ensuing signal emission, $\mathrm{V}_b\left(t\right)$ describes the potential of the cell axon, and the variable τ_b controls the shape of $\mathrm{F}_b\left(t\right)$, which leads to a low respective high pass filter by subtracting $\mathrm{F}_b\left(t\right)$ in $\frac{\partial \mathrm{V}_b(t)}{\partial t}$.

Figure 6.1.9 shows the response of a simulated transient and sustained ON-BC on different stimuli. With suitable parametrization ($\tau_b = 0.25$ for transient and $\tau_b = 2.0$ for sustained BCs and $\sigma_b = 0.15$), the shape of the results coincides with the measurements of Cleland et al. on cats [160], Awatramani et al. on tiger salamander [161], and Zhao et al. on mice [162]. Figure 6.1.10 shows an example of simulated spatio-temporal signal flow. It is well recognizable how the potential through the transient bipolar cells decreases more slowly compared to Figure 6.1.6, and 6.1.8.

Retinal Ganglion Cells (RGCs) modulate the signal arising from the BCs in two ways: By interconnecting ON-BCs and OFF-BCs, they form the difference between center and periphery (receptive field), as modeled by the signal propagation likelihood $P_g\left(\theta|V_B\right)$. Depending on the emerging input potential, RGCs also modulate the signal temporally by so-called spikes. Spikes are short voltage peaks whose frequency varies according to the applied potential. A distinction is made between phasic and tonic ganglion cells. Tonic RGCs emit spikes only when the voltage is applied, whereas phasic cells also emit spikes at rest. This different behavior is supposedly less an intrinsic property of the RGC but probably results from the selective connection to the BCs [154], [160]–[162].

Lankheet et al. proposed a model for the simulation of spike generation by RGCs back in 1988 [272]. In 1990 Fohlmeister et al. published a detailed simulation of the current flows based on the ion channels Na^+, Ca^{2+}, K^+, and the resulting spikes based on studies on tiger salamanders [273], [274], which was revised by Sikora et al. in 2004 [275] and Lei Wang et al. in 2014 [276].

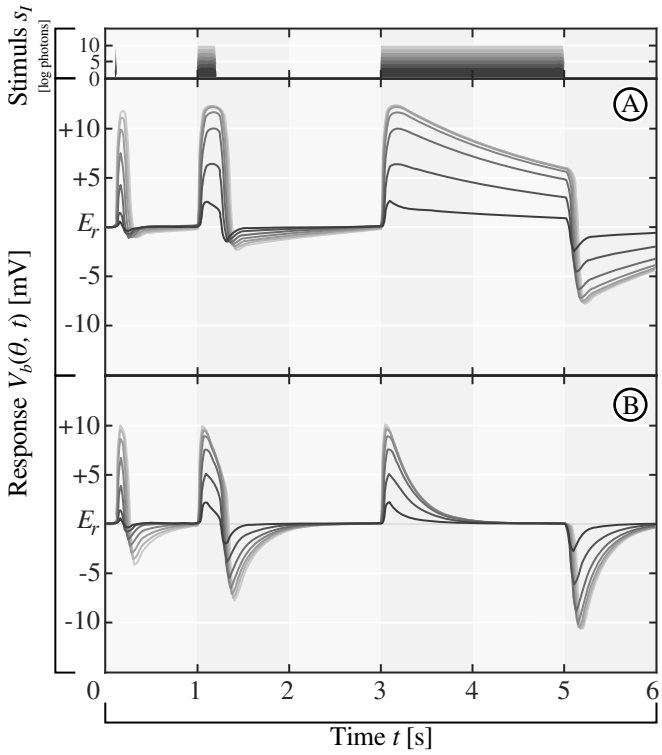


Fig. 6.1.9: Simulation result of an ON-BC. Section (A) shows the simulated response of a sustained BC on a stimulus of 10 ms and 100 ms light flash, with 3 to 10 log photons at 560 nm wavelength each, and $\tau_b = 0.25$. (B) shows the simulated response of a transient BC, at the same stimulus and $\tau_b = 2.0$. Both simulations show a continuous reduction of membrane voltage after the first peak due to the hyperpolarization, with a transient BC the membrane voltage converging significantly faster to zero, whereas the sustained BC converges towards a value above zero until the stimulus stops.

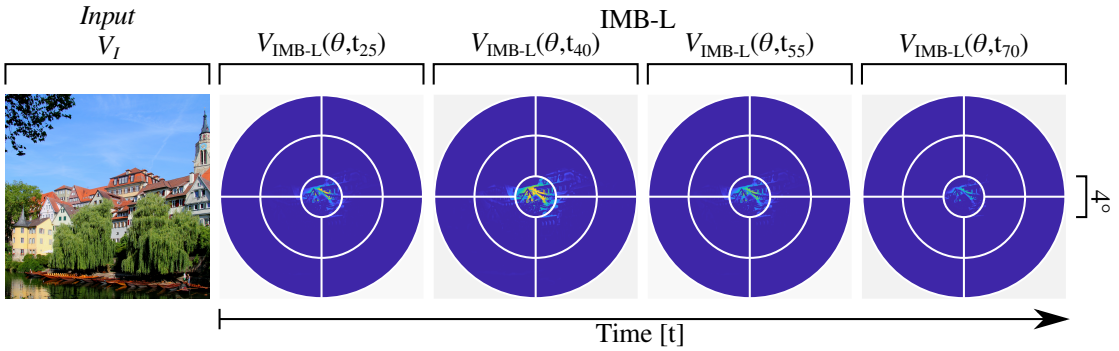


Fig. 6.1.10: The combined spatial-temporal signal of transient and sustained IMB-L BCs.

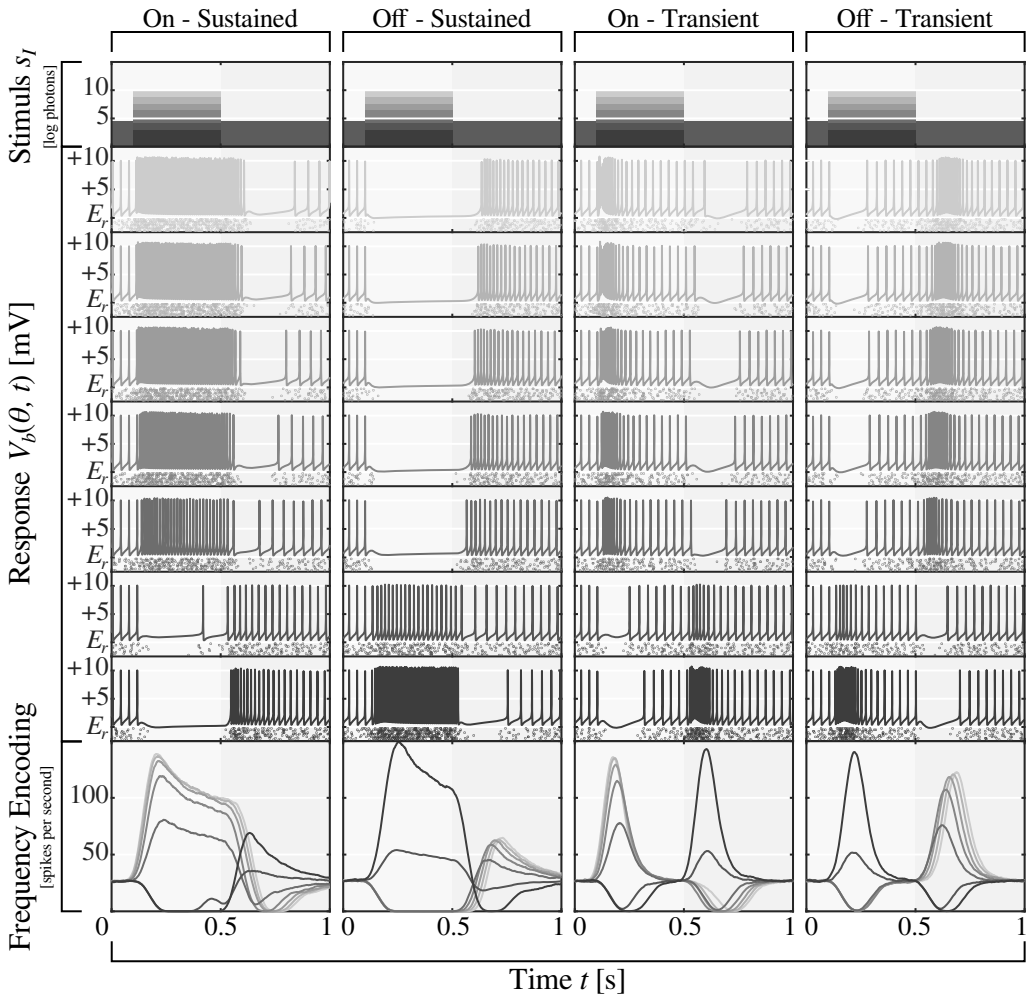


Fig. 6.1.11: Simulated RGC responses. The first column shows the simulation of a RGC with an on-center and an off-periphery, center and periphery is linked to a single sustained BC, which in turn is linked to a single L-cone in the center and an M-cone for the periphery. The second column shows the simulation of an off instead of an on-center. And the last two columns showing the results using transient BCs instead of sustained BCs. The first row indicates the simulated intensity of the stimulus that the photoreceptors are exposed to. A light incidence of 3 to 10 log photons with a wavelength of 564 nm (maximum sensitivity of the L-cones) was simulated. The following seven rows (10 to 3 log photons) indicate the potential change relative to the resting potential of the RGC. The scattered points below the potential curve indicate spike events raised by the stimulus of the first 10 of 1000 simulations. The last row shows the frequency of the generated spikes, estimated over a window of 80 ms and 1000 simulations.

$$
\begin{aligned}
C_m \cdot \frac{\partial V_g(\theta,t)}{\partial t} = & \frac{\partial \text{stim}_g\left(V_g(\theta,t),\theta,t\right)}{\partial t} + \xi(t) \\
& - \frac{\partial \text{Na}^+\left(V_g(\theta,t)\right)}{\partial t} - \frac{\partial \text{K}^+\left(V_g(\theta,t)\right)}{\partial t} \\
& - \frac{\partial \text{Ca}^{2+}\left(V_g(\theta,t)\right)}{\partial t} - \frac{\partial \text{A}\left(V_g(\theta,t)\right)}{\partial t} \\
& - \frac{\partial \text{K}^+\text{Ca}^{2+}\left(V_g(\theta,t)\right)}{\partial t} - \frac{\partial \text{L}\left(V_g(\theta,t)\right)}{\partial t}
\end{aligned}
, \tag{6.21}
$$

where $\text{stim}_g(\theta,t)$ is the potential arising from the stimulus, and $\xi(t)$ is Gaussian white noise with a mean of zero. The current of the ion channels are modeled by $\text{Na}_g^+(\theta,t)$, ..., $\text{K}^+\text{Ca}_g^{2+}\left(V_g(\theta,t),t\right)$, where $\text{A}\left(V_g(\theta,t)\right)$ denotes the inactivating K^+ channel, and $\text{K}^+\text{Ca}_g^{2+}\left(V_g(\theta,t),t\right)$ describes the Ca^{2+} activated K^+ currents. $\text{L}_g\left(V_g(\theta,t),t\right)$ is the leakage current, which is set to zero here. According to Fohlmeister et al., the ion channels can be modeled as follows:

$$
\begin{aligned}
\frac{\partial \text{Na}^+(V_g)}{\partial t} &= g_{\text{Na}^+} \cdot m^3 \cdot h \cdot (V_g - V_{\text{Na}^+}) \\
\frac{\partial \text{K}^+(V_g)}{\partial t} &= g_{\text{K}^+} \cdot n^4 \cdot (V_g - V_{\text{K}^+}) \\
\frac{\partial \text{Ca}^{2+}(V_g)}{\partial t} &= g_{\text{Ca}^{2+}} \cdot c^4 \cdot \left(V_g - \frac{R \cdot T}{2 \cdot F} \cdot \ln\left(\frac{\text{Ca}_\text{e}^{2+}}{\text{Ca}_\text{i}^{2+}(V_g)}\right)\right), \\
\frac{\partial \text{A}(V_g)}{\partial t} &= g_\text{A} \cdot a^3 \cdot b \cdot (V_g - V_{\text{K}^+}) \\
\frac{\partial \text{K}^+\text{Ca}^{2+}(V_g)}{\partial t} &= g_{\text{K}^+\text{Ca}^{2+}} \cdot \frac{\text{Ca}_\text{i}^{2+}(V_g)^2}{\text{Ca}_\text{i}^{2+}(V_g)^2 + \text{Ca}_\text{diss}^{2+\,2}}
\end{aligned}
\tag{6.22}
$$

where $g_{\text{Na}^+}, \ldots, g_{\text{K}^+\text{Ca}^{2+}}$ are the rate constants of the voltage-gated ion channels, followed by the state variables m, h, n, c, a and b calculated by the Hodgkin-Huxely first order kinetic equation: $\frac{\partial x}{\partial t} = \alpha_x - (\alpha_x + \beta_x)$, where x represents one of the state variables. The parameters where fitted by Fohlmeister et al. as followed [274]:

$$
\begin{array}{lll}
g_{\text{Na}^+} = 50\,\text{mS}\,\text{cm}^{-2} & \alpha_m = \frac{-0.6\cdot(V_g+30)}{e^{-0.1\cdot(V_g+30)}-1} & \beta_m = 20 \cdot e^{-0.0556\cdot(V_g+55)} \\
 & \alpha_h = -0.4 \cdot e^{-0.05\cdot(V_g+50)} & \beta_h = \frac{6}{e^{-0.1\cdot(V_g+20)}+1} \\
g_{\text{K}^+} = 12\,\text{mS}\,\text{cm}^{-2} & \alpha_n = \frac{-0.02\cdot(V_g+40)}{e^{-0.1\cdot(V_g+40)}-1} & \beta_n = 0.4 \cdot e^{-0.0125\cdot(V_g+50)} \\
g_{\text{Ca}^{2+}} = 2.2\,\text{mS}\,\text{cm}^{-2} & \alpha_c = \frac{-0.3\cdot(V_g+13)}{e^{-0.1\cdot(V_g+13)}-1} & \beta_c = 10 \cdot e^{-0.0556\cdot(V_g+38)} \\
g_\text{A} = 36\,\text{mS}\,\text{cm}^{-2} & \alpha_a = \frac{-0.006\cdot(V_g+90)}{e^{-0.1\cdot(V_g+90)}-1} & \beta_a = 0.1 \cdot e^{-0.1\cdot(V_g+30)} \\
 & \alpha_b = -0.04 \cdot e^{-0.05\cdot(V_g+70)} & \beta_b = \frac{0.6}{e^{-0.1\cdot(V_g+40)}+1} \\
g_{\text{K}^+\text{Ca}^{2+}} = 0.05\,\text{mS}\,\text{cm}^{-2} &
\end{array}
, \tag{6.23}
$$

where the membrane potential V_g is denoted herein mV. The equalization potential of the

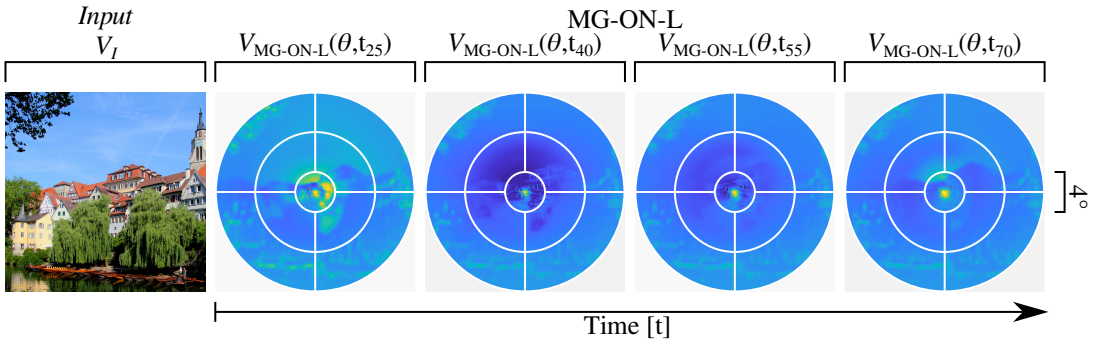


Fig. 6.1.12: Simulated frequency of MG-ON-L RGCs.

Na^+ channel is set to $V_{Na^+} = 35\,\text{mV}$ and the K^+ channel to $V_{K^+} = -75\,\text{mV}$, whereas the reflux of the Ca^{2+} channel is time-dependent and is modeled using the Nernst equation:

$$\frac{\partial \text{Ca}_\text{i}^{2+}(V_g)}{\partial t} = 1.5 \cdot 10^{-5} \cdot \frac{\partial \text{Ca}^{2+}(V_g)}{\partial t} - 0.02 \cdot \left(\text{Ca}_\text{i}^{2+}(V_g) - 10^{-4}\right). \tag{6.24}$$

Please refer to the publications of Fohlmeister et al. [273], [274] for a detailed breakdown of the parameters.

The potential arises at the retinal location θ depends on the local linkage properties of the RGCs g, which is described by the signal flow probability $P_g(V_B|\theta)$:

$$\frac{\partial \text{stim}_g(V_g,\theta,t)}{\partial t} = e^{10\cdot P_g(V_B|\theta)-2.687} - (V_g + 65), \tag{6.25}$$

Figure 6.1.11 shows the simulated spike encoding of sustained and transient RGCs. However, since the handling of spikes is delicate for spatial retinal visualizations and successive calculation, V_g is interpreted as the frequency of spikes. Figure 6.1.12 shows the simulated ganglion response for MG-ON-L cells on different timestamps, where Figure 6.1.13 provides the simulation output of all RGCs with an ON center.

6.1.1.3 Implementation

The implementation of the introduced model is structured in four successive processing steps: pre-processing, neural computing, post-processing, and classification. Figure 6.1.14 illustrates this processing pipeline as feed forward-network.

The proposed model is implemented using C/C++ and the excessive use of Cuda (there is also a pure host code version available). Additionally, the provided interfaces allow using the library in Matlab (MEX wrapper) or in Simulink (S-functions). The source code is freely available at: `https://atreus.informatik.uni-tuebingen.de/geislerd/visuallperceptionmodel`

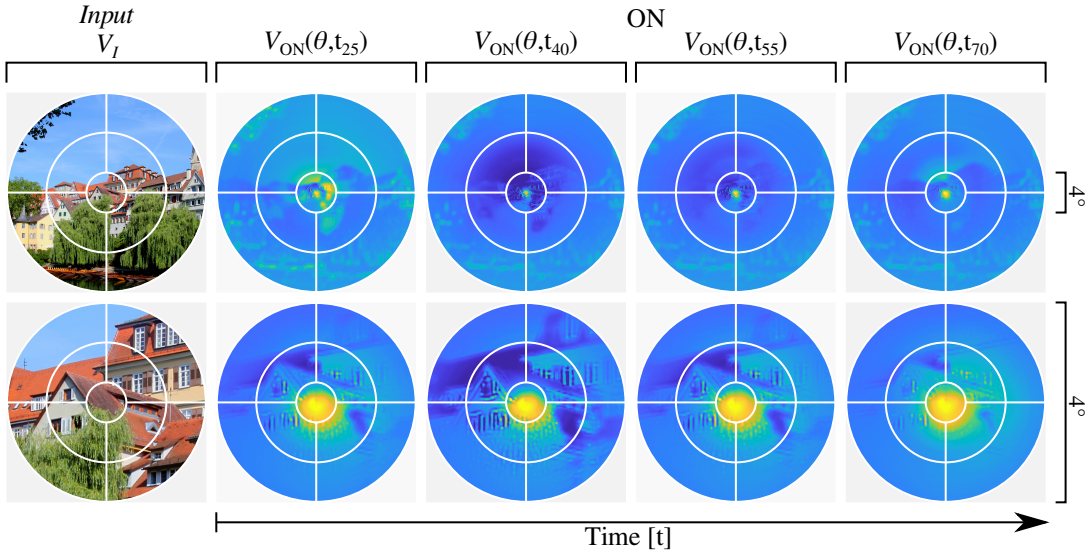


Fig. 6.1.13: Shows the pooled RGC response. The bottom row shows the ganglion response zoomed into the foveal region.

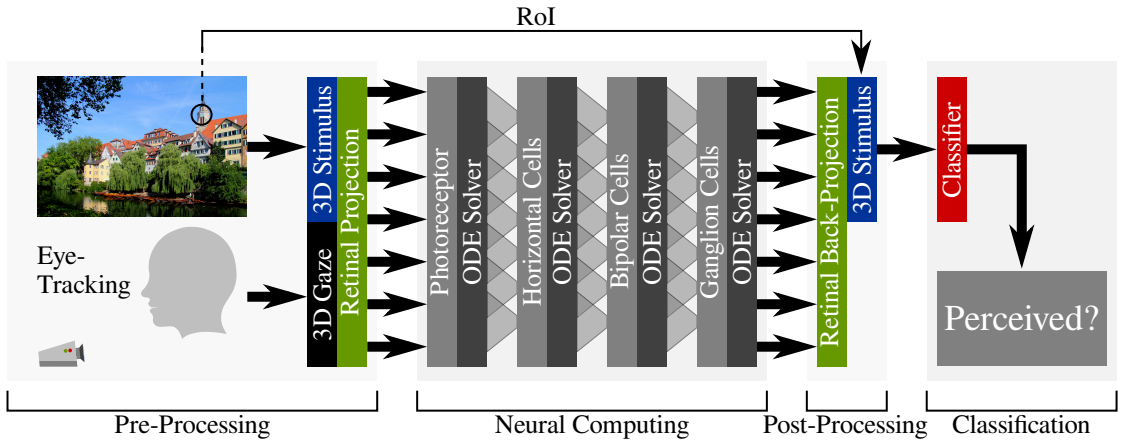


Fig. 6.1.14: Framework architecture of the proposed retinal model. Starting with a reprocessing step to uniform the input data, ready to get projected into the user's retinal coordinate system using an eye-tracking system. The neural computing consists of four specialized layers respective to the described retinal processing steps in Section 6.1.1.1, followed by an ODE solver to modulate the temporal signal as described in Section 6.1.1.2. In the post-processing step, the predicted retinal activation is back-projected to the scene and mapped to the input 3D stimulus. Finally, the ROI is classified using an SVM as perceived or not.

Pre-processing: This step aims to harmonize the data structures. The implementation requires a projection of the scene into a retinal coordinate system using an eye-tracking signal. In the following, it is assumed that the scene content is available as a set of RGB voxels, as often found in RGB-D camera systems or camera and LiDAR combinations. Furthermore, the user's visual axis is assumed to be given as a ray, with an origin in the eyeball and a gaze direction, as introduced in Section 2.1. This line of sight representation is achievable using state-of-the-art remote eye-tracking systems such as Smart-Eye or Tobii Pro (refer Section 2.3 for details). However, it is also possible to use head-mounted eye-tracking systems (even low-cost ones) and a 3D scene reconstruction using ArUco markers, similar to Section 6.2.2.1 and 2.2.

Then the retinal eccentricity of each voxel regarding the detected line of sight is calculated and interpolated into a grid. Subsequently, the resulting retinal scene image is passed to the neural processing step.

Neural Computing: This step calculates the spatial signal flow and temporal modulation throughout the different retinal layers. It implements the actual bio-physiological model as a series of four successive processing layers: photoreceptors, horizontal cells, bipolar cells, and ganglion cells. Each layer consists of a probabilistic signal flow prediction (see Section 6.1.1.1), and a successive ordinary differential equation solver (see Section 6.1.1.2). Subsequently, the calculated output of the ganglion cell layer is passed to the post-processing step.

Post-Processing: Most applications require the intensity of the retinal activation, arose by a particular scene area, e.g., perceivability of a particular traffic situation, hazardous objects, notification, warning, etc. Hence, the post-processing step back-projects the response of the ganglion to the 3D stimulus. Subsequently, the retinal response of RoI is easily accessible.

Classification: The following Section 6.1.2.2, demonstrates a final classification step. Therefore, a SVM assesses the retinal response to predict whether the user perceives a particular scene region. Even more exciting is the challenge of whether the retinal response is capable of predicting the attentive perception in the future. However, this step is not part of the actual retinal model. Rather it uses its output as an example application.

6.1.2 Evaluation

The following sections present evaluation results of the introduced retinal processing model in terms of its predictiveness of neural activity and the user's visual perception of distinct scene objects. Thereby, the focus is to evaluate the model to predict a peripheral object perception and a subsequent saccade to it. Section 6.1.2.1 introduces an extensive driving simulation study with a sophisticated task design. The study provides the ground truth user perception, annotated by manual user feedback (keystroke), yet exact timing determination due to processing the recorded line of sight and eye movements analysis. Section 6.1.2.2

presents the user's perception evaluation and results, including a saccade prediction, where a performed saccade towards a scene object is predicted even before the user executed it.

6.1.2.1 Dataset

The evaluation dataset was recorded on a $147° \times 39°$ canvas setup. Each of the three canvas had a size of $3.05\,\text{m} \times 1.89\,\text{m}$ and was tilted at $151°$ vertically. The canvas displayed a grid (150 horizontally, 25 vertically) of different objects (triangles, circles, squares) analog to the study of Treisman et al. in their *Feature integration theory* [104]. However, random animations such as wobbling, color changes, or out fading were applied to object to induce temporal dynamics. The participating 19 users were instructed to seek those object animations and acknowledged their perception due to a keystroke. The user's keystroke provides the ground truth about whether an object animation was perceived. Not acknowledged object animations were considered as not perceived. Simultaneously, the users' line of sight was recorded using a *SmartEye Pro* system. Based on the recorded line of sight, a post-processing step analyzed the earliest saccade towards an acknowledged perceived object animation, as the timestamp of perception. In this way, a total of $1,010$ not perceived and 839 perceived (total $1,849$) object animations were collected.

6.1.2.2 Results

The proposed retina model was applied to each of the recorded object modification. Subsequently, the MG's response peak and its corresponding gaze eccentricity were extracted as feature (B) (as illustrated in Figure 6.1.15). From this point, the gaze signal was traced backward to identify the corresponding location of the saccade to the object. As a second feature (B), the highest PG's response peak until the previously detected saccade to the object was extracted.

Figure 6.1.16 shows the distribution of the sampled MG and PG response, as well as the minimal gaze distance to the object. In the next step, an SVM (Gaussian kernel) was trained using Matlab's Classification Toolbox to predict whether the user perceives an object animation. Figure 6.1.17 shows the respective confusion matrices, using both features (A) and (B) or individual. It is apparent that the success rate by using the MG response is very high. However, in this setup, a similar success rate ($0.4\,\%$ less accuracy) is achievable just using the minimal gaze distance – without applying a retina model at all. This is mainly attributable to the general high object's saliency on the black background. Additionally, the users were like to focusing directly on a found object animation. Yet, this evaluation does not include temporal aspects.

Figure 6.1.17 also indicates that the PGs achieve a quite high accuracy as well ($84.7\,\%$). This is remarkable since these cells mainly populate the peripheral retinal region. Moreover, the respective feature (A) was extracted to always precede the saccade to the object (as illustrated in Figure 6.1.15). In other words, the classifier predicts with an accuracy of $84.7\,\%$ whether a future saccade will target the object. Figure 6.1.18 shows that the model predicts the peripheral perception of an object animation 100 ms in advance of the saccade to the object, with an accuracy of almost $70\,\%$.

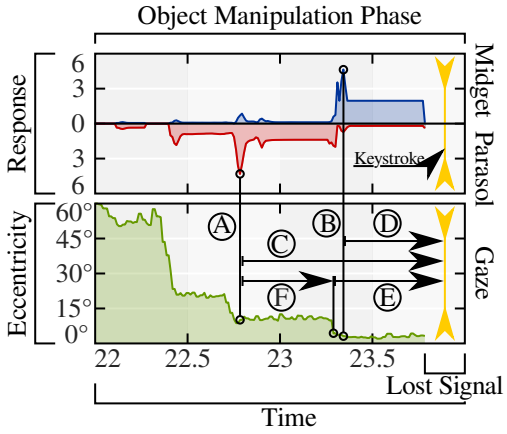
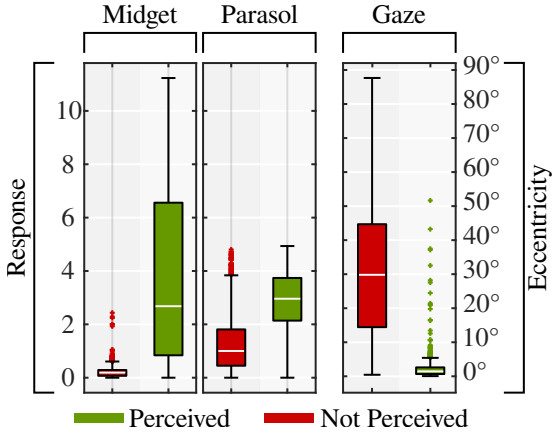


Fig. 6.1.15: Shows a representative case of the retinal response to a perceived scene object manipulation. The blue area plot indicates the MG response (predominantly foveal). The red area plot indicates the PG response (predominantly peripheral). The lower plot shows in green the distance of the foveal axis to the examined scene object. The yellow bar marks the time where the user has acknowledged the perception. It is quite noticeable that the object first raises a peripheral amplitude. Then, after about 0.5 ms, the saccade to the object follows, and foveal perception increases. (A) and (B) show the features used for the classification: The respective RGC response peak value combined with the eccentricity between the gaze vector and object. (C) to (E) indicates the period from PG peak, MG peak, and saccade to the user acknowledgment. (F) indicates the period between PG peak and saccade.

Fig. 6.1.16: Shows the maximum retinal response of MGs and PGs, as well as the minimum gaze distance for perceived and not perceived object manipulations.



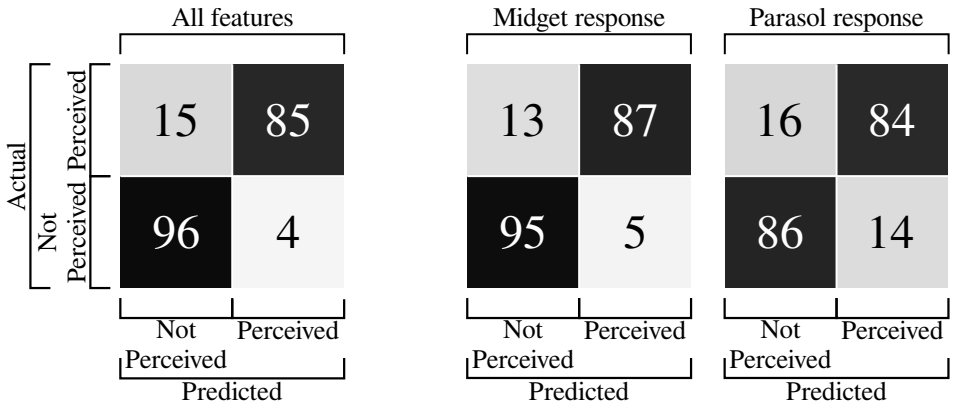


Fig. 6.1.17: Achieved classifications rate in percent. The first confusion matrix uses the maximum response of both MG and PG RGCs (see Figure 6.1.15 features (A) and (B)). The second and third confusion matrix uses only the maximum response of one RGC type.

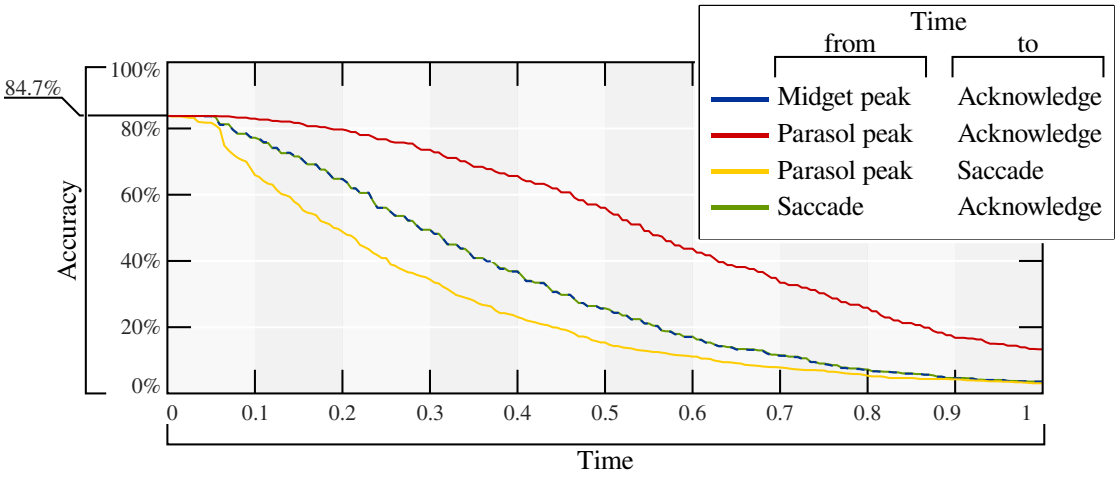


Fig. 6.1.18: Shows the classification accuracy under the condition that the prediction was made a certain time before the saccade to the object.

The experiment was conducted by Bickerdt and Geisler et al. and is published with a comprehensive design description and data analysis at Journal of Eye Movement Research (JEMR) [277].

6.1.3 Limitations

The presented model processes a given stimulus as accurately as possible in terms of bio-physiological findings. However, the human retinal blueprint is still a very active research domain and is continuously extended and supplemented with new research findings. Even though the proposed model covers a substantial part of retinal signal modulation, it must be assumed that some signal pathways are not adequately represented. In particular, the role of the ACs is only sparsely specified and not considered in the model. However, this work invites researchers to tackle this lack and to incorporate their findings into the presented model in order to develop an even more comprehensive model of the retinal signal processing and in-depth understanding of how humans perceive the scene.

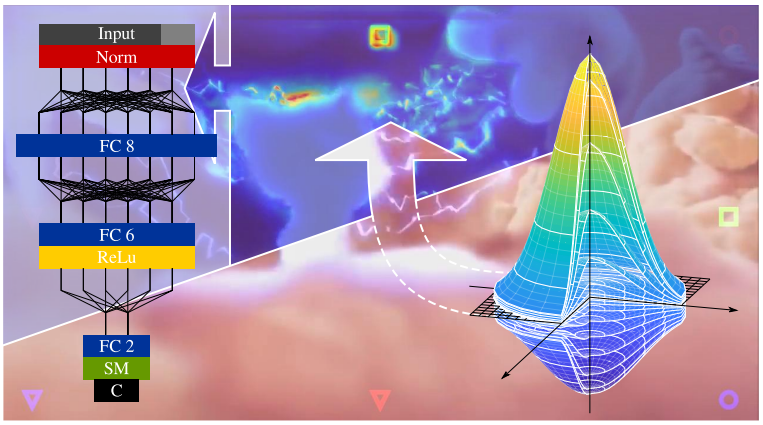
The evaluation shows that the high requirements on the eye-tracking accuracy, as well as the sweeping computing capacity, are challenging. Both may require to process the retina model in a low – insufficient resolution, in order to compensate for tracking noise or low computing capacities. However, the rapid development of vision-based eye-tracking systems in recent years promises that even high-precision setups will be affordable to a broad community soon. Moreover, the next Section 6.2 introduces a computing efficient retinal model by using functional replications of the neural signal modulation.

6.1.4 Conclusion

This section presented a model for accurate bio-physiologically driven prediction of retinal neuronal response. It is based on physiological and neurological findings and replicates accurately as possible the retinal signal flow and modulation. Thus, the proposed model enables researchers to analyze precisely how specific scene contents are retinal processed and allows inferences as to whether a particular stimulus is perceptible to a spectator. Moreover, the evaluation reveals that the model is able to predict the peripheral perception, up to whether a saccade to a particular object will be performed, using the assessed peripheral retinal neuronal activity.

Besides a accurate replication of the signal flow, the presented model extends the state-of-the-art of an assessment over a large field of view. This provides the research community with a tool that allows the examination of foveal visual perception as well as peripheral perception. More important, it combines research findings from bio-physiological studies and neuroscience and formalizes them in a unique mathematical model. This makes them available to a broad community at the application level and encourages the interdisciplinary transfer of current research findings.

6.2 Functional model



The previous Section 6.1 introduced a bio-physiological retinal model, enabling a remarkable accuracy in predicting the user's peripheral perception. Yet, its application is severely restricted in many application domains due to its extensive hardware requirements and execution runtime. Thus, e.g., in the automotive context, realtime evaluation is a fundamental prerequisite. Simultaneously, even today, most vehicles provide limited computing capacities. However, applications in the field of driver monitoring are of particular significance. For instance, a car may determine whether the driver is aware of a detected hazardous situation through its environment sensors combined with the driver's line of sight. Alternatively, in the case of highly automated driving, a car may predict whether the driver has perceived traffic-critical scene objects before the manual control is handover. Navigation systems may adapt their HUD displays to ensure not to superimpose critical scene contents, yet still perceptible to the driver. The number of possible applications already only in this context is extremely versatile.

The following sections are supported by a prior publication from Geisler et al. at the Journal of Neurocomputing (NEUCOMP) in 2020 [4]. The presented approach address the computing capacity challenge of the bio-physiological model presented in Section 6.1, by introducing a hybrid approach. Similar to the gaze-driven saliency approach presented in Section 5.2, a functional replication of the retinal stimulus processing is used instead of the complex and computationally intensive neural computation implemented in Section 6.1. However, the parameterization of the model is bio-physiologically reasoned and extends over the user's entire visual field.

Section 6.2.1 first describes the proposed model in detail. A comprehensive evaluation is then provided in Section 6.2.2 along with the gaze-driven GBVS approach from Section 5.2, the original GBVS, the Itti & Koch saliency map, and a gaze-only classification of perception. The final Sections 6.2.3 and 6.2.4 provide the limitations and the conclusion, respectively.

6.2.1 Method

Physiological studies show that the receptive fields formed by the RGCs process the underlying stimulus in a manner similar to a high-pass filter. In fact, this effect is often modeled using a rectified Differential of Gaussian ($\mathcal{D}o\mathcal{G}$), since it sufficiently reflects the spatial properties of RGCs with a color channel in the center and the opponent color channel in the surround (center-surround differences) [154], [278], [279]. With increasing eccentricity from the fovea, the size of the receptive fields increases, leading to less detailed contrast perception in peripheral vision [280] (see Figure 2.4.7). In addition to spatial contrast, temporal contrast is formed and amplified (mainly by the underlying BCs and HCs), which causes particular sensitivity to movement and suddenly appearing visual stimuli (e.g., light flashes) [281].

The proposed filter models the functionality of the receptive fields formed by the retinal RGCs as a rectified high pass filter in the spatio-temporal domain. Taking into account locally varying retinal capabilities, the filter is parameterized according to its eccentricity from the visual axis. In this way, the proposed filter is applied to the five most relevant signal modulation paths for visual perception: the chromatic red-green, green-red, and blue-yellow channels, as well as the achromatic diffuse and luminance contrast channel. Subsequently, the resulting stimulus prediction of the retinal feature extraction is weighted according to the local biological manifestation of the respective modulation path. Finally, the stimulus of the local retinal area is evaluated in terms of the perceptibility of the scene object.

The prediction of the incoming stimulus requires the projection of the scene content into a retinal coordinate system whose center represents the origin of the visual axis – the fovea. In turn, the projection requires reliable tracking of the user's visual axis with simultaneous capturing of the scene content. This step depends strongly on the used sensors and their data provision and may need to be adapted to the respective application environment. Section 2.1 described various fundamental techniques for state-of-the-art head-mounted, yet also remote eye-tracking to facilitate these requirements. Moreover, the experiment Section 6.2.2.1 describes a sufficient setup, realized using a low-cost state-of-the-art head-mounted eye-tracking system and a desktop monitor scenario.

The following methodology assumes that the scene content is available as a set of RGB voxels, as often found in RGB-D camera systems or camera and LiDAR combinations:

$$\mathbf{S} = \left(\overbrace{\begin{matrix} s_{0,x} & s_{0,y} & s_{0,z} \\ & \dots & \\ s_{k,x} & s_{k,y} & s_{k,z} \end{matrix}}^{s_{i,xyz}} \overbrace{\begin{matrix} s_{0,r} & s_{0,g} & s_{0,b} \\ & \dots & \\ s_{k,r} & s_{k,g} & s_{k,b} \end{matrix}}^{s_{i,rgb}} \right) \begin{matrix} \left.\begin{matrix} \\ \\ \end{matrix}\right\} \begin{matrix} s_0 \\ \vdots \\ s_k \end{matrix} \end{matrix} , \tag{6.26}$$

where $s_{i,x}$, $s_{i,y}$, and $s_{i,z}$ are the coordinates of the i-th voxel s_i in the set $\mathbf{S}$, where x represents the horizontal, y the vertical, and z is the depth axis. The coordinate vector will also be referred to as $s_{i,xyz}$ or as the set of all coordinates $\mathbf{S}_{xyz}$. Analogously, $s_{i,r}$, $s_{i,g}$, and $s_{i,b}$ describe the RGB values, and $s_{i,rgb}$ the RGB vector of the i-th voxel. The set of all RGB values is referred to as $\mathbf{S}_{rgb}$. Moreover, it is assumed that the corresponding visual axis is

given as a ray in the following form:

$$g = \overbrace{\begin{pmatrix} g_{\mathrm{o},x} \\ g_{\mathrm{o},y} \\ g_{\mathrm{o},z} \end{pmatrix}}^{g_\mathrm{o}} + g_r \cdot \overbrace{\begin{pmatrix} g_{\mathrm{d},x} \\ g_{\mathrm{d},y} \\ g_{\mathrm{d},z} \end{pmatrix}}^{g_\mathrm{d}}, \tag{6.27}$$

where g_o is the origin of the gaze in the scene coordinate system, g_r is the scalar distance from the gaze origin to the scene, and g_d is the normalized gaze direction, as introduced in Section 2.1.

The following six consecutive steps describe the preparation and retinal projection of the recorded data ①, the prediction of the incoming stimulus ②, modeling of the five signal paths ③, application of the receptive field filter ④, the weighting of the extracted retinal stimulus ⑤, and finally the prediction of perceptibility ⑥.

Step ① First, the scene is transformed into the retinal coordinate system. Therefore, it is assumed that the eyeball resembles a spherical shape, where the retina forms the back and wraps around the optical center:

$$\mathbf{S}_{\theta\phi} = \overset{\triangleleft}{\mathrm{ret}} \left(\overbrace{\frac{\mathbf{S}_{xyz} - g_\mathrm{o}}{\|\mathbf{S}_{xyz} - g_\mathrm{o}\|} - g_\mathrm{d}}^{\hat{\mathbf{S}}_{xyz}} \right), \tag{6.28}$$

where the function $\overset{\triangleleft}{\mathrm{ret}}\left(\mathbf{S}_{xyz}\right)$ converts the cartesian deviation from the gaze ray g to the scene voxels $\mathbf{S}$ into the retinal coordinate system $\hat{\mathbf{S}}_{\theta\phi}$ defined by the axis θ and ϕ, representing the horizontal and vertical eccentricity respectively:

$$\overset{\triangleleft}{\mathrm{ret}}\left(\hat{\mathbf{S}}_{xyz}\right) = \begin{cases} \left(\sin^{-1}\left(\frac{\hat{\mathbf{S}}_x}{\|\hat{\mathbf{S}}_{xyz}\|}\right), \sin^{-1}\left(\frac{\hat{\mathbf{S}}_y}{\|\hat{\mathbf{S}}_{xyz}\|}\right)\right), & \text{if } \hat{\mathbf{S}}_z \geq 0 \\ \emptyset, & \text{else } [1] \end{cases}. \tag{6.29}$$

Finally, the scattered scene data $\mathbf{S}$ is interpolated into a grid of $\pm\pi$ horizontally and vertically:

$$\left(\mathbf{S}_{\theta\phi} \middle| \mathbf{S}_{rgb}\right) \xrightarrow[\text{2D interpolation}]{-\pi \leq (\theta,\phi) \leq +\pi} \left(\hat{\mathbf{S}}_{\theta\phi} \middle| \hat{\mathbf{S}}_{rgb}\right). \tag{6.30}$$

This step is not mathematically mandatory, but it facilitates the subsequent convolution with the receptive field filter in step ④ as it simplifies the memory layout and parallelization.

Step ② After projecting the scene onto the retinal surface $\hat{\mathbf{S}}_{\theta\phi}$, the second step calculates the induced stimulus affecting the photoreceptors. Figure 2.4.4 shows the sensitivity of the individual photoreceptors according to the wavelength. Since the scene is usually not known in its full light spectrum but is usually recorded by RGB cameras, the actual stimulus can only be roughly estimated. Therefore, the RGB colors of the stimulus $\hat{\mathbf{S}}_{rgb}$ are transferred into the LMS color space, which depicts the sensitivity of the three different cone types: L,

[1] Voxel coordinate is behind the eyeball.

M, and S (long-, medium-, and short-wavelength), and represents the effective color space to humans [282], [283]. The stimulus affecting the rods is approximated by the CIE scotopic luminosity function (see Figure 2.4.4), as these are quite similar regarding their spectral sensitivity [284]:

$$\hat{\mathbf{S}}_{LMSR} = \hat{\mathbf{S}}_{rgb} \cdot \underbrace{\left(\overbrace{\begin{matrix} 0.6172 \\ 0.7797 \\ 0.0412 \end{matrix}}^{\text{L}} \overbrace{\begin{matrix} 0.0671 \\ 0.9681 \\ 0.0691 \end{matrix}}^{\text{M}} \overbrace{\begin{matrix} -0.0018 \\ 0.0192 \\ 1.0259 \end{matrix}}^{\text{S}} \overbrace{\begin{matrix} -0.3278 \\ 0.8603 \\ 0.5388 \end{matrix}}^{\text{R}} \right)}_{\mathbf{T}} \begin{matrix} \text{r} \\ \text{g} \\ \text{b} \end{matrix} . \tag{6.31}$$

The values of **T**, represent the computed linear transformation from the RGB Color Matching Function (CMF) from Stiles & Burch to the LMS CMF from Stockman & Sharpe and the CIE 1951 scotopic luminosity function based on Crawford & Wald (column R) [284]–[288].

Step ③ Spatial visual perception is essentially based on the contrast formation of the different cone and rod color spaces. The receptive fields form a center-surround difference over the opponent chromatic color spaces L-M (red-green), M-L (green-red), and S-LM (blue-yellow), as well as the achromatic color space LMSR-LMSR (diffuse) and R-R (luminance). In addition, a temporal difference is formed by sustained and transient modulation of the photoreceptor signals [159], [161]. In order to reproduce the temporal signal modulation in the next step, the individual color channels $\mathbf{S}_L, \ldots, \mathbf{S}_R$ are written into a respective ring buffer $\boldsymbol{B}_{cc}^{L}, \ldots, \boldsymbol{B}_{cc}^{R}$ of size t_{max} with each newly recorded stimulus. Thus, each of the ring buffers forms a 3D tensor, with the axes representing the horizontal (θ), vertical (ϕ), and temporal (t) development of the stimulus. The opponent color buffer belonging to $\boldsymbol{B}_{cc}$ is in the following referred to as $\boldsymbol{B}_{\overline{cc}}$. Thus, the opponent color spaces are denoted by the following buffers:

Opponent color	Color channel buffer $\boldsymbol{B}_{cc}$	Opponent color channel buffer $\boldsymbol{B}_{\overline{cc}}$
L-M	$\boldsymbol{B}_{cc}^{L}$	$\boldsymbol{B}_{cc}^{M}$
M-L	$\boldsymbol{B}_{cc}^{M}$	$\boldsymbol{B}_{cc}^{L}$
S-ML	$\boldsymbol{B}_{cc}^{S}$	$0.5 \cdot (\boldsymbol{B}_{cc}^{L} + \cdot \boldsymbol{B}_{cc}^{M})$
LMSR-LMSR	$0.25 \cdot (\boldsymbol{B}_{cc}^{L} + \boldsymbol{B}_{cc}^{M} + \boldsymbol{B}_{cc}^{S} + \boldsymbol{B}_{cc}^{R})$	$0.25 \cdot (\boldsymbol{B}_{cc}^{L} + \boldsymbol{B}_{cc}^{M} + \boldsymbol{B}_{cc}^{S} + \boldsymbol{B}_{cc}^{R})$
R-R	$\boldsymbol{B}_{cc}^{R}$	$\boldsymbol{B}_{cc}^{R}$

Step ④ The function of the receptive fields formed by the RGCs can be described as a rectified high-pass filter over the two opposing color channels $\boldsymbol{B}_{cc}$ and $\boldsymbol{B}_{\overline{cc}}$. Thereby, the difference is calculated between the high passed color channel to its closer surroundings of the opposite color channel (center-surround difference) [278]. This spatial behavior is often modeled as a $\mathcal{D}o\mathcal{G}(\sigma_{\text{c}}, \sigma_{\text{s}})$ (difference of gaussian) filter, where the two parameters $\sigma_{\text{c}} < \sigma_{\text{s}}$ represent the radii of the inner and outer area of the receptive field [110].

We adopted this approach by adding an additional dimension to describe the temporal

contrast and adjusting the scaling of the $\mathcal{DoG}$ to the eccentricity to the fovea:

$$\begin{aligned}\hat{\mathbf{L}}_{cc}(\theta,\phi) = &\sum_{\hat{\theta},\hat{\phi},t} \boldsymbol{B}_{cc}\left(\theta+\hat{\theta},\phi+\hat{\phi},t\right)\cdot \boldsymbol{F}_{cc}\left(\hat{\theta},\hat{\phi},t\right)+\\ &\sum_{\hat{\theta},\hat{\phi},t} \boldsymbol{B}_{\overline{cc}}\left(\theta+\hat{\theta},\phi+\hat{\phi},t\right)\cdot \boldsymbol{F}_{\overline{cc}}\left(\hat{\theta},\hat{\phi},t\right)\end{aligned}, \tag{6.32}$$

where $\hat{\theta}$ and $\hat{\phi}$ are retinal coordinates ranging from $-\pi$ to π. t is the temporal domain of the buffers $\boldsymbol{B}_{cc}$ and $\boldsymbol{B}_{\overline{cc}}$ ranging from the most recent entry t_0 to the last recent $t_{\max}$ (according to the example in Figure 6.2.2 and 6.2.1, $t \in \{t_0,\ldots,t_{-12}\}$). The functions $\boldsymbol{F}_{cc}(\theta,\phi,t)$ and $\boldsymbol{F}_{\overline{cc}}(\theta,\phi,t)$ are Gaussian-like 3D kernels, as illustrated in Figure 6.2.1:

$$\hat{\boldsymbol{F}}_{cc}(\theta,\phi,t) = e^{t-t_0}\cdot \mathcal{N}_{\mu=0}\left(\sqrt{\theta^2+\phi^2},\sigma_{cc}\left(\sqrt{\theta^2+\phi^2}\right)\right), \tag{6.33}$$

and

$$\hat{\boldsymbol{F}}_{\overline{cc}}(\theta,\phi,t) = -e^{t_{\max}-t}\cdot \mathcal{N}_{\mu=0}\left(\sqrt{\theta^2+\phi^2},\sigma_{\overline{cc}}\left(\sqrt{\theta^2+\phi^2}\right)\right), \tag{6.34}$$

where $\mathcal{N}_{\mu=0}(x,\sigma^2)$ returns the value of a gaussian normal distribution at x with a mean of $\mu=0$ and the given variance σ^2. The leading factor in the equation leads to an exponential weighting of the time series, whereas the latest stimulus in the center of the receptive field ($\hat{\boldsymbol{F}}_{cc}$) and the last stimulus in the periphery ($\hat{\boldsymbol{F}}_{\overline{cc}}$) are emphasized (please note that $t_0 > t_{\max}$). The scaling functions σ_{cc} and $\sigma_{\overline{cc}}$ describe the radius of the receptive field in relation to the retinal eccentricity.

Figure 2.4.7 shows the increase of the field size by increasing eccentricity to the fovea for different ganglion cell types. While midget ganglion cells form mainly the chromatic contrasts L-M and M-L, the S-ML contrast is mainly formed by the larger bistratified ganglion cells [289], [290]. Achromatic contrasts are processed by a parasol and wide-field ganglion cells [291]. $\sigma_{\overline{cc}}$ is therefore chosen proportional to the related curve in Figure 2.4.7, assuming that $\sim$3.6 mm eccentricity corresponds to about 1° deviation in the visual field of view [262]. The inner radius σ_{cc} is set to half of the outer one: $\sigma_{cc}(x) = 0.5\cdot\sigma_{\overline{cc}}(x)$.

Finally, both kernels are normalized; thus the sum of $\boldsymbol{F}_{cc}$ and $\boldsymbol{F}_{\overline{cc}}$ results in 1:

$$\boldsymbol{F}_{cc}(\theta,\phi,t) = \frac{\hat{\boldsymbol{F}}_{cc}(\theta,\phi,t)}{\sum_{\theta,\phi,t}\hat{\boldsymbol{F}}_{cc}(\theta,\phi,t)}, \tag{6.35}$$

where the normalization of $\hat{\boldsymbol{F}}_{\overline{cc}}$ is defined analogously.

Thus obtained temporal opponent color channel contrasts $\hat{\mathbf{L}}_{L-M},\ldots,\hat{\mathbf{L}}_{R-R}$ represent the estimated stimulus of the opponent color channels of the retina induced in a time-series $t_0,\ldots,t_{\max}$ of $\mathbf{S}_{rgb}$. Figure 6.2.3 provides an example of the temporal opponent color channel after 5ms fixation.

Step ⑤ Figure 2.4.6 shows the relative frequency of the four most frequent RGC types: MG, BG, PG, and WG. The foveal area is dominated by the chromatic MGs, which form the

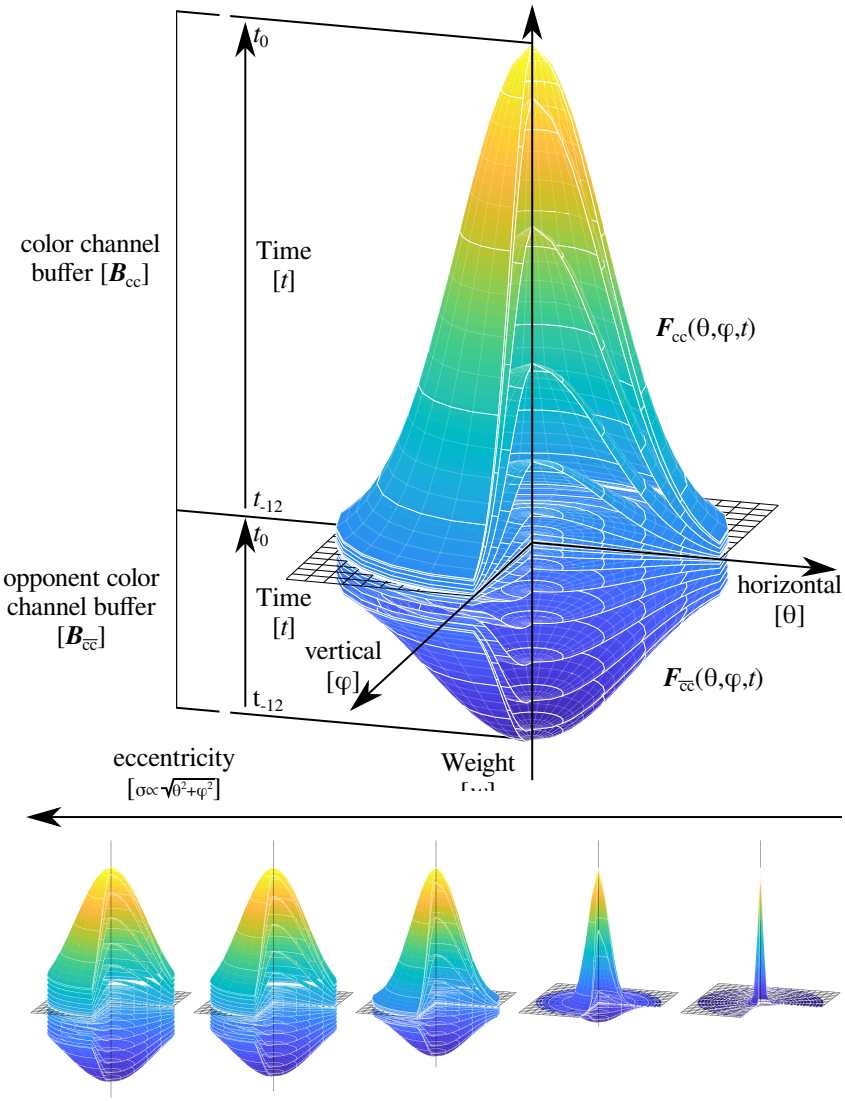


Fig. 6.2.1: Illustration of the convolution spatio-temporal receptive field convolution kernel $\boldsymbol{F}_{cc}$ and the corresponding opposite $\boldsymbol{F}_{\overline{cc}}$. As the eccentricity increases, the shape of the kernel becomes broader, resulting in a larger receptive field.

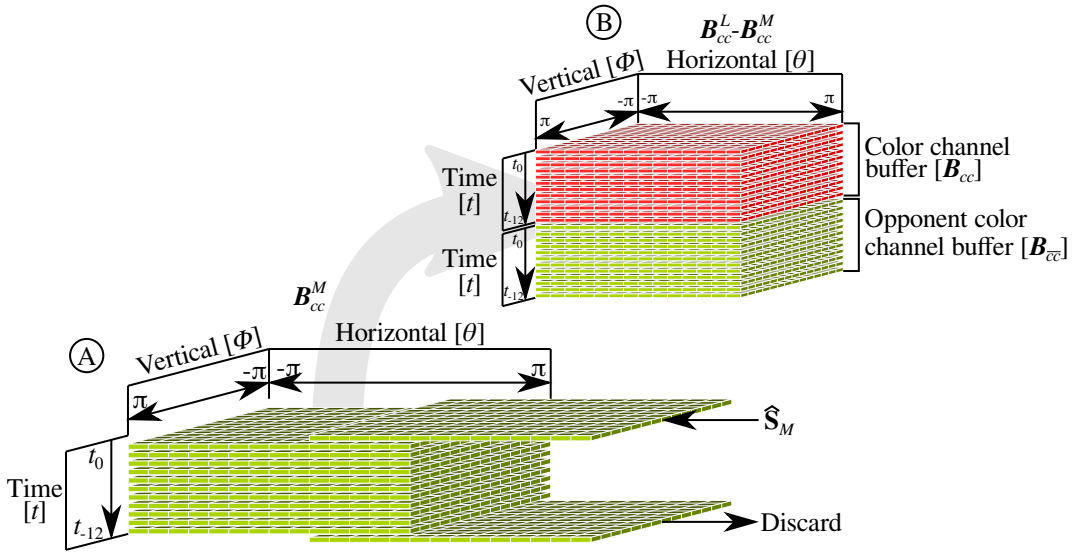


Fig. 6.2.2: Structure of the buffers of the opponent color channels. Part (A) shows the three dimensions of the $\mathbf{S}_M$ color channel ring buffer. The new scene $\mathbf{S}_M$ forms the top element (t_0) of the ring buffer $\boldsymbol{B}^M_{cc}$. The last element is discarded. Part (B) shows the concatenation of two color channel buffers $\boldsymbol{B}^L_{cc}$ and $\boldsymbol{B}^M_{cc}$, to prepare the convolution with the receptive field core F^{L-M}_{cc}.

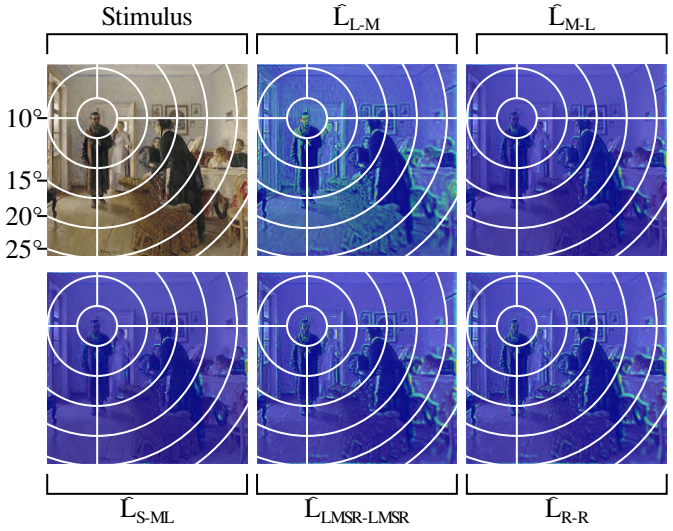


Fig. 6.2.3: Temporal opponent color channels $\hat{\mathbf{L}}_{L-M}, \ldots, \hat{\mathbf{L}}_{R-R}$ on the stimulus *Unexpected visitors* from Ilja Jefimowitsch Repin. The acuity decreases with increasing eccentricity due to larger RGCs in the periphery.

opponent color channels L-M and M-L. The BGs, responsible for the opponent color channel S-ML are located in the outer area of the fovea and extends far into the periphery. The achromatic PGs and WGs, on the other hand, occur mainly in the peripheral field of vision, leading to increasing achromatic perception by increasing eccentricity to the fovea [145], [150], [292].

This understanding leads to different perceptual capacities depending on the affected retinal area. In order to include this into the perception model, the retinal stimuli $\hat{\mathbf{L}}_{L-M}$, $\ldots$, $\hat{\mathbf{L}}_{R-R}$ calculated in step ④ are weighted according to its eccentricity and the frequency of the corresponding RGC type:

$$\mathbf{L}_{cc}(\theta, \phi) = w_{cc}\left(\sqrt{\theta^2 + \phi^2}\right) \cdot \hat{\mathbf{L}}_{cc}(\theta, \phi), \tag{6.36}$$

where $\mathbf{L}_{cc}(\theta, \phi)$ and $\hat{\mathbf{L}}_{cc}(\theta, \phi)$ return the estimated stimulus at the retinal position (θ, ϕ). The weighting $w_{cc}(d)$ returns the corresponding curve value at eccentricity d in Figure 2.4.6. For the color channels L-M and M-L, the value must be divided by 2, since both are formed by the MGs.

Step ⑥ If no task-specific weighting is known or determinable, a naive approach is to combine the different stimulus layers $\mathbf{L}_{L-M}, \ldots, \mathbf{L}_{R-R}$ over an averaged sum in order to obtain a final prediction of retinal perception $\mathbf{I}$:

$$\mathbf{I} = \mathbf{L}_{L-M} + \mathbf{L}_{M-L} + \mathbf{L}_{S-ML} + \mathbf{L}_{LMSR-LMSR} + \mathbf{L}_{R-R}. \tag{6.37}$$

The perceptibility of a certain scene object can then be derived directly from $\mathbf{I}$ and classified with a threshold value if necessary.

However, training a small, fully connected neural network to perform the weighting and combination of the stimuli maps $\mathbf{L}_{L-M}, \ldots, \mathbf{L}_{R-R}$, enables a subsequent prediction of whether a scene object was recognizable. Therefore, the retinal area on which the examined scene object is projected is cut out from the maps $\mathbf{L}_{L-M}, \ldots, \mathbf{L}_{R-R}$ over the examined time period $t - n, \ldots, t$ and denoted as patches $\mathbf{P}_{cc}^{(t-n)}, \ldots, \mathbf{P}_{cc}^{(t)}$. Then, the maximum spatial and temporal stimuli are determined from the obtained series of patches and are used together with its minimal retinal eccentricity as the classifier input feature vector $\mathbf{f}$:

$$\mathbf{f} = \left(\max\left(\mathbf{P}_{L-M}^{(t-n)}, \ldots, \mathbf{P}_{L-M}^{(t)}\right), \ldots, \max\left(\mathbf{P}_{R-R}^{(t-n)}, \ldots, \mathbf{P}_{R-R}^{(t)}\right), \min\left(\mathbf{P}_{\theta\phi}^{(t-n)}, \ldots, \mathbf{P}_{\theta\phi}^{(t)}\right)\right), \tag{6.38}$$

where the functions $\max(\ldots)$ and $\min(\ldots)$ extract the extreme values over the spatial and temporal domain. $\mathbf{P}_{\theta\phi}$ provides the retinal coordinates of the examined patch $\mathbf{P}$ analogous to Equation 6.28; thus the last row of $\mathbf{f}$ indicates the shortest distance from the fovea to the object over the examined time span.

6.2.2 Evaluation

The evaluation comprises the presented retinal model, yet also the saliency-based approach, presented in Section 5.2. The following Section 6.2.2.1 introduces the conducted experiment

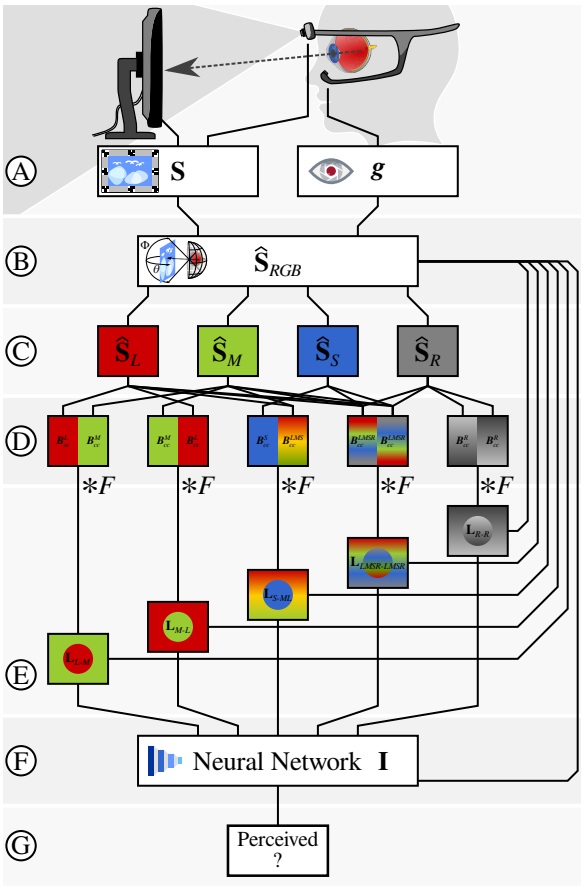


Fig. 6.2.4: The workflow of the proposed filtering pipeline of the visual stimulus: Section (A) shows the experimental setup with a monitor and a head-mounted eye-tracker. In section (B), the stimulus is projected onto the retina and interpolated. The stimulus of the individual photoreceptors is estimated in Section (C). Then, Sections (D) and (E) form the opponent color channels and estimate the response of the receptive fields with the proposed filter F_{cc} and $F_{\overline{cc}}$, as shown in Figure 6.2.1. Finally, in Sections (F) and (G), the affected retinal sections are extracted from the response maps of section (E). Subsequently, a NN combines, weights, and evaluates the patches with respect to the perceptibility of the examined scene object.

to collect the evaluation data. Section 6.2.2.2 presents and discusses the evaluation results.

6.2.2.1 Experiment

The following experiment was initially designed as proof of concept and evaluation for the proposed approach. However, the experiment was also used to collect data for training and evaluation of the weighting and classification step ⑥ regarding a visual search task. Ultimately, this experiment also demonstrates how the necessary prerequisites can be created using a desktop setup.

In order to track the user's visual axis, the participants were equipped with a head-mounted eye-tracker from PupilLabs (2nd version) [53]. The recording and evaluation of the eye videos were performed by the open-source software EyeRec using the pupil detection algorithm Purest (see Section 2.2) [58]. The mapping of the pupil position to the field camera was estimated by a polynomial function using the CalibMe calibration [10], [17], [69].

The experiment scene consisted of a $\sim$6 min animation short movie [293] (Partly Cloudy). During the presentation, eight random HUD objects were superimposed (using addition in the RGB color space) on the scene borders, as shown in Figure 6.2.5 and A.2. Superimposed objects were either circles, squares, or triangles, which were randomly colored red, green, or blue. During the movie playback, random animations were applied to the objects, and multiple animations to multiple objects could be shown simultaneously. Object animations were a repetitive color change (red to green and vice versa or blue to yellow and vice versa), a wobbly movement (movement along an ∞-molded trajectory), or a pulsating fade-in and fade-out of the object, and lasted no more than 2 s. Between two animations of the same object was at least a 1 s pause. The participants were instructed to acknowledge all perceived changes of the objects with a corresponding keystroke.

In order to prevent continuous scanning of the objects, the participants were asked to count all present creatures in the scene (there are a lot of them). Accordingly, superimposed HUD objects on the screen must be primarily tracked in the peripheral field of view, since the visual focus is strongly bound to the counting task and storyline of the movie. At the same time, the perception of the objects can be very challenging if, e.g., the background is very bright or has a similar coloration as the HUD objects. From previous experiments and literature, it can be expected that the visual perception of the object animations is classified into three difficulty levels:

1st The movements of the objects can be perceived in the entire visual field. Additionally, at a suitable background, they may address all of the five considered opponent color channels, which makes them particularly easy to perceive.

2nd The object color change animations were performed in the LAB color space in order to minimize the number of noticeable intensity changes. Thus, the color transition is primarily perceptible in the affected opponent color channels. In addition, there are no spatial changes during the color transition, meaning a smaller part of the photoreceptors is stimulated compared to movements.

3rd The fading out of the object is most difficult to perceive. Although it also creates an intensity contrast and is therefore perceivable across the entire field of view: The

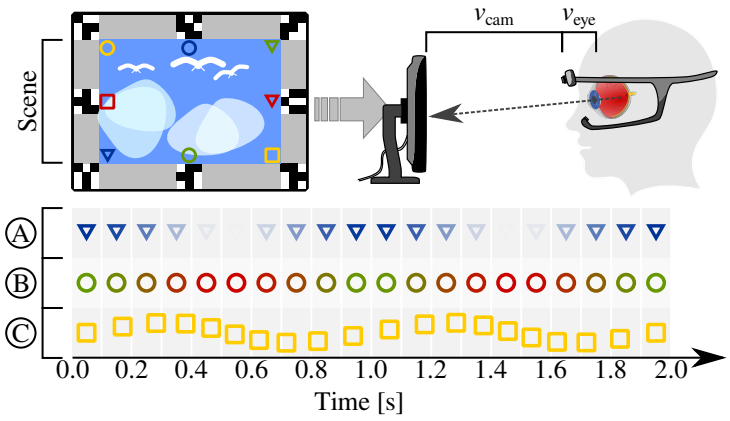


Fig. 6.2.5: Experiment design: The scene is surrounded by nine ArUCO markers for localization of the field camera to the screen. The eight randomly shaped and colored objects are drawn at the border area at the scene. The columns (A)-(C) shows the sequence of the three different animation types: (A) the fade-in and -out of the object, (B) the color change from green to red, (C) the sequence of a motion animation.

contrast is usually low, depending on the background. In addition, it is often difficult to tell whether it is a brightening of the background or a fading out of the object.

A total of nine participants with no known eye impairments were recorded. While viewing the scene, a total of 571 object animations were presented, of which 247 (43.26 %) were acknowledged as seen with a keystroke. Table 6.2 shows the distribution of the collected data. As expected, movements were perceived very frequently, whereas color transitions and the fading in and out of the object were often not detected. Also, it was apparent that changes to blue objects were perceived less frequently than red or green objects. On the one hand, this effect is probably due to the fact that the movie contains a lot of blue-tinted sky scenes. On the other hand, the comparatively small number of S-cones (only 8 %-10 % of cone population [294], [295]) may also be decisive. This circumstance would also explain why the blue objects had significantly more perceptions at color transitions compared to the out- and in-fading of the object. While the more frequent L- and M-cones are more stimulated during the color transition to yellow, the fading out of the object at a blue toned background does not induce an appreciable stimulus. This notion is also consistent with the statements of some participants that they sometimes needed to observe the fade-out and fade-in of the objects for some time to determine whether it was animation or an intensity change of the background video.

Simultaneous to the recording of the perception rates of object manipulations, the corresponding 3D gaze direction was estimated. ArUco markers around the scene (see Figure 6.2.5) were used to map the 2D gaze signal on the field camera to the scene on the screen using a camera undistortion and a homography matrix by solving the following Equation [74],

Objects and animations frequency

		Red	Green	Blue	Animation frequency
Ⓐ	Fade-in/-out	11.7%	14.0%	8.93%	34.7%
Ⓑ	Color transition	9.63%	13.3%	9.11%	32.0%
Ⓒ	Motion	9.81%	12.1%	11.4%	33.3%
	Color frequency	31.2%	39.4%	29.4%	

Relative frequency of perceived objects and animations frequency

*averaged values

		Red	Green	Blue	Animation frequency*
Ⓐ	Fade-in/-out	29.9%	32.5%	17.6%	26.7%
Ⓑ	Color transition	27.3%	31.6%	26.9%	28.6%
Ⓒ	Motion	83.9%	84.1%	52.3%	73.4%
	Color frequency*	47.0%	49.4%	32.3%	

Tab. 6.2: The upper table shows the relative frequency of randomly applied animations to objects, broken down by object color and type of animation. The lower table shows the relative frequency that an object was marked as seen while the animation was applied to the subject.

[296], [297]:

$$\overbrace{\begin{pmatrix} x_{\text{screen},j} \\ y_{\text{screen},j} \\ 1 \end{pmatrix}}^{w_{\text{screen},j}} = H \overbrace{\begin{pmatrix} x_{\text{cam},j} \\ y_{\text{cam},j} \\ 1 \end{pmatrix}}^{w_{\text{cam},j}}, \tag{6.39}$$

where $w_{\text{screen},j}$ is the known screen coordinate of the j-th ArUco marker, and $w_{\text{cam},j}$ is the corresponding detection in the field camera. H is the 3×3 homography matrix, which is determined by at least four of the eight ArUco markers using an Singular Value Decomposition (SVD) [298]. H is then used to map the 2D gaze points from the camera coordinate system to the screen coordinate system. In addition, the field camera position v_{cam} is estimated in 3D relative to the screen using PnP:

$$\overbrace{\begin{pmatrix} x_{\text{cam},j} \\ y_{\text{cam},j} \\ 1 \end{pmatrix}}^{w_{\text{cam},j}} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \left(R_{\text{cam}} \middle| v_{\text{cam}}\right) \overbrace{\begin{pmatrix} x_{\text{world},j} \\ y_{\text{world},j} \\ 0 \\ 1 \end{pmatrix}}^{w_{\text{world},j}}, \tag{6.40}$$

where $w_{\text{cam},j}$ is again the observed ArUco marker position in the camera. (f_x, f_y) is the focal length of the camera, and (c_x, c_y) is the principal point (the skew parameter is assumed

to be zero here). $\left(R_{\text{cam}} \middle| v_{\text{cam}}\right)$ indicates the projection matrix of the camera, where R_{cam} is the 3×3 rotation matrix, and v_{cam} the 3D translation of the camera. Finally, $w_{\text{world},j}$ is the world coordinate of the j-th ArUco marker on the screen.

The offset from the camera to the eye was roughly estimated by $v_{\text{eye}} = R_{\text{cam}} \cdot (-25, -10, -40)^\top$mm. Since the estimated 2D gaze signal already combines both eyes together to a single output, a cyclopic model was assumed: Whose eye is located at the nose root. The stimulus $\mathbf{S}$ contains all pixels of the displayed scene, where $s_{i,x}$ and $s_{i,y}$ correspond to the (x, y) mm-coordinate of the i-th pixel on the screen ($s_{i,z} = 0$ mm), and $s_{i,rgb}$ contains the RGB value of the pixel. The 3D gaze direction g is derived from the relative eye position $g_{\text{o}} = v_{\text{cam}} + v_{\text{eye}}$ to the screen and the corresponding pixel coordinate of the homographic mapped 2D scene gaze.

6.2.2.2 Results

The presented model was applied to the recorded experiment scene using the tracked line of sight to predict whether an object animation was perceived by the subject. During an applied animation of one of the objects, the temporal and spatial maximum of the estimated stimulus $\mathbf{L}_{L-M}, \ldots, \mathbf{L}_{R-R}$ within a patch padded with 50 px around the object was sampled (as described in step ⑥): $\mathrm{P}_{L-M}, \ldots, \mathrm{P}_{R-R}$. Additionally, the minimum gaze signal distance to the object's border $\mathrm{P}_{\theta\phi}$ was calculated. The intention is that the additional gaze signal may enable the classifier to weight the five retinal feature channels differently according to their eccentricity in the user's field of view. Overall, this led to a six-element feature vector for each applied object animation: $\mathbf{f} = (\mathrm{P}_{\theta\phi}, \mathrm{P}_{L-M}, \ldots, \mathrm{P}_{R-R})$

Moreover, three additional data sets based on the response of the saliency map from Itti & Koch, GBVS (see Section 2.4.1), and the gaze-driven GBVS approach of Geisler et al. (see Section 5.2), were recorded analogously [2], [110], [240]:

Itti & Koch: replicates the retinal receptive fields, quite similar to the presented approach. However, Itti & Koch's intention is not to predict the visual perception in an online fashion, but to predict the visual perception in a free viewing task without time constraints [110]. The assumption is that a scene area raising a strong retinal response is more frequently fixated and is therefore perceived more intensely. Nevertheless, the predicted saliency combined with the tracked line of sight may provide information about whether a scene object was perceivable by the user. Therefore, the saliency map for each video frame was processed by using the implementation of the Simpsal library provided by J. Harel, with the recommended settings for videos (but with 128 px instead of 32 px internal resolution) [248]. As above, the maximum response over time was then calculated in a patch around the object animation and form. Together with the minimum distance of the gaze to the object, this leads to a two-element feature vector: $\mathbf{f} = (\mathrm{P}_{\theta\phi}, \mathrm{P}_{\text{Saliency}})$.

GBVS: The saliency map approach of Harel et al. distinguishes from Itti & Koch in using a Markov model instead of center-surround differences. The Markov model's edge weights depend on the spatial and feature distance between individual pixels. Thus, pixels outstanding from their neighborhood get empowered by the Markov model

on a random walk [240]. The two-element feature vector was calculated analog to the Itti & Koch approach (even using the same toolbox). However, the internal resolution was set to default at 32 px, since running the Markov model is much more resource-intensive than Itti & Koch.

Gaze-driven GBVS: Geisler et al. recently introduced a modified GBVS that incorporates the gaze signal into the Markov model to visualize a user's stimulus perception rate. Therefore, the gaze point encodes the initial Markov state vector. Thus, the GBVS emphasize especially salient regions around the gaze point. Moreover, as the Markov state vector becomes predominantly sparse, the approach is way less resource-intensive compared to the original GBVS [2].

Finally, a fifth data set, only consisting of the minimum distance of the gaze to the object border, provided the benchmark baseline: $\mathbf{f} = (\mathrm{P}_{\theta\phi})$. This complies with the most popular method to determine whether an object was perceived by a user: A tracked line of sight is traced and evaluated with regard to boundary boxes in the scene. The appendix Figure A.2 shows an instance of the extracted visual feature vectors from the experimental.

To assess and qualify the validity of the extracted scene features by the proposed model, the classifiers were trained on each of the five data sets. The classification target was to predict whether an object animation was perceived by the user. The user's keystroke (or no keystroke) whenever an object animation was perceived, provided the ground truth perception. All keystrokes from the beginning of the animation to a maximum of 1 s after the end of the animation were counted to compensate for possible delays by the reaction time. The resulting data set comprises 247 (43 %) positive and 324 (56 %) negative data points (see also Table 6.2).

Since there was no prior knowledge about the loss landscape, five different classifiers with a total of 23 preset configurations from Matlab's 2018a Classification Learner were trained and evaluated:

Decision trees conjugate many weak classifiers (often simple comparators) as nodes in a tree structure into a powerful one [299]. Three different configurations of decision trees were trained: Fine, medium, and coarse tree with a maximum of 100, 20, and 10 decisions. Furthermore, three ensembled trees were trained using Bagging, Boosting, and RUSBoost with a maximum of 570, 30, and 20 decisions, respectively [300]–[302].

Discriminant functions assign scores to each observation. Based on the score, the group membership and boundaries between the groups are determined. Three different discriminant classifiers were trained: linear, quadratic, and subspace ensembled [303].

Logistic regression models the class probability, assuming a binomial distribution [304].

Support Vector Machine (SVM) classify observations according to a trained hyperplane [305]. The SVM was trained with a linear, quadratic, and cubic kernel, as well as a Gaussian kernel (RBF) using the three different scaling parameters: fine ($\sqrt{P}/4$), medium ($\sqrt{P}$), coarse ($4 \cdot \sqrt{P}$).

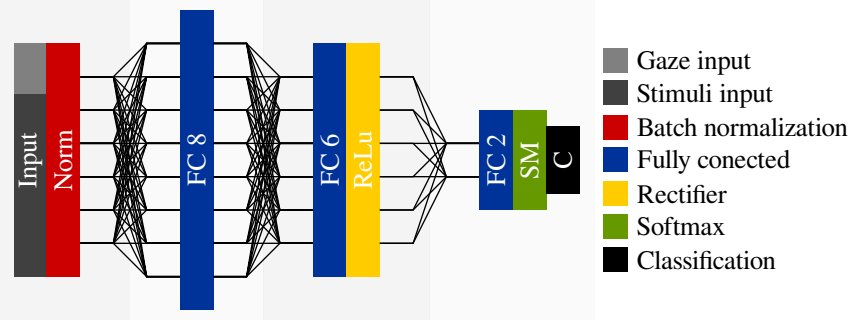


Fig. 6.2.6: Architecture of the NN that achieved the highest accuracy in the experiment.

k-Nearest Neighbor (kNN) classifies the observation by looking up the k nearest neighbors and counting their class according to the distance to the observation [306]. Altogether six different configurations were trained: $k = 1$ (fine), $k = 10$ (medium), and $k = 100$ (coarse) neighbors using equal weighting as well as a cosine, cubic, and linear distance weightings for $k = 10$ neighbors was trained.

All classifiers were evaluated using a 10-fold cross-validation scheme, using a balanced data set with 50 % of positive and negative samples, leading to 49 evaluation vs. 445 training samples.

In addition, a small, fully connected NN was trained. Figure 6.2.6 illustrates the best performing neural network architecture found during a heuristic search. It provides 16 neurons, divided into a 8-, 6-, and 2-neuron Fully Connected Layers (FCs) (108 weights total). The first two FCs abstract and weight the several features followed by a rectification; the last FC layer calculates a pseudo-likelihood, followed by a softmax, and a classification layer. It was found that more extensive networks tended to overfit on the training data, which may also be due to the comparatively low amount of training data. Smaller networks, on the other hand, seemed to be less accurate.

The training of the neural network did not take place within the same toolbox as the other classifiers, and therefore differs slightly. Like the other classifiers, the performance was also determined by cross-validation, but with a total of 500 folds. Therefore, for each of the folds, a randomly balanced set from the experiment data was chosen (494 samples). This set was further divided into 10 % evaluation data (49 samples), 10 % validation data (49 samples), and 80 % training data (396 samples). The training data were then used to start 4 parallel training sessions. After finishing the training, each of the training sessions was validated against the validation dataset. If a network reaches less than 75 % accuracy, it was discarded, and the training was repeated up to five times. If, after five attempts, still no training run achieved a higher accuracy on the validation dataset above 75 %, the best of the five trials was selected. Finally, the one with the highest accuracy of the four training sessions was chosen as the fold result and evaluated using the evaluation data set. The evaluation resulted in an average accuracy of 81.5 % with a true positive rate of 76.2 % and a true negative rate of 86.9 %, which moderately outperforms the other classifiers.

The appendix Figure A.3 shows that, apart from the neural network, the decision trees

		Red	Green	Blue	Accuracy
Ⓐ	Fade-in/-out	84.3 %	75.8 %	72.8 %	77.6 %
Ⓑ	Color transition	78.9 %	86.0 %	86.3 %	83.7 %
Ⓒ	Motion	81.4 %	91.4 %	77.0 %	83.3 %
	Accuracy	81.5 %	84.4 %	78.7 %	81.5 %

	Red	Green	Blue	True positive
Fade-in/-out	74.3 %	72.1 %	66.4 %	71.0 %
Color transition	69.1 %	87.6 %	98.7 %	85.1 %
Motion	72.3 %	88.2 %	56.7 %	72.4 %
True positive	71.9 %	82.6 %	74.0 %	76.2 %

	Red	Green	Blue	True negative
Fade-in/-out	94.2 %	79.4 %	79.2 %	84.3 %
Color transition	88.7 %	84.4 %	73.9 %	82.3 %
Motion	90.5 %	94.6 %	97.3 %	94.1 %
True negative	91.1 %	86.1 %	83.5 %	86.9 %

Tab. 6.3: The upper table show the accuracy of the small NN classifier broken down by object color and type of animation. The left and right tables shows the true positive and true negative detection rate, respectively.

show a similarly good performance, with an average accuracy of up to 80.4 %. Whereby, these methods are characterized by a higher true positive rate. Table 6.3 breaks down the recognition rates by object color and animation type. The recognition rate is relatively evenly distributed across the different objects and animations, with a slight tendency towards a higher true negative rate. However, it stands out that the accuracy of blue objects is significantly lower, especially if they are faded-out as animation. This effect is probably because the background of the scene is often blue. When the object is faded out, there is hardly any additional contrast that can be captured by the model, which leads to a lower true positive rate. The human retina is certainly much more sensitive in this respect. It can also be assumed that the participants become aware that they need to pay particular attention to such objects and thus become especially sensitive to such effects.

Figure 6.2.7 indicates the classification accuracy of the neural network based on the five different data sets. The features extracted using the Itti & Koch model achieve maximum accuracy of 72 % (small NN). The mere Gaze classification achieves an accuracy of almost 67 %. Followed by the classification accuracy based on GBVS features and gaze. The poor performance of GBVS is mainly due to the very low runtime reasoned internal resolution. Figure A.2 shows that the superimposed objects are too small to impact the saliency map significantly. Yet, the actual superiority of the GBVS reveals in the gaze driven GBVS derivative of Geisler et al. With more than 77 %, the gaze-driven GBVS outperforms the gaze-only approach at > 10 %. However, the highest accuracy of > 81 % is achieved based on the proposed features in Section 6.2.1. These features reflect the local retinal perceptual capabilities more differentiated and accurate than the gaze-driven GBVS approach.

Compared to it, the presented approach shows an improvement in the accuracy of 7.96 %. This is probably due to the fact that the Itti & Koch model does not account for temporal contrasts or the different perceptual abilities of the retina. For instance, a red-green contrast always appears as salient in the Itti & Koch model, regardless of whether it is mapped onto a retinal area where only achromatic contrasts are processed.

To evaluate how strong the benefit of the additional stimuli features is compared to

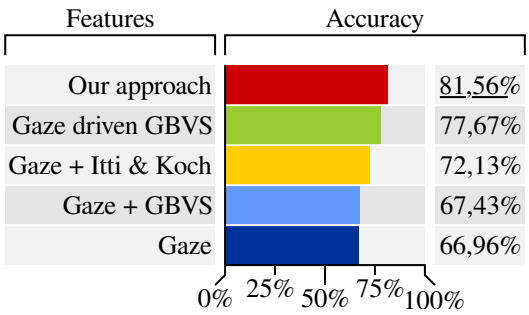


Fig. 6.2.7: Classification accuracy of the proposed NN based on different input features. The bottom row shows the accuracy based on the mere gaze signal (blue). The next two rows show the accuracy by gaze and saliency, calculated using Itti & Koch (light blue) and GBVS (yellow), respectively [110], [240]. The second top row indicates the classification accuracy using the gaze driven GBVS from Geisler et al. (green) [2]. The top row provides the accuracy using the proposed features (red).

gaze-only classification, all the classifiers were trained and re-evaluated by only using the minimal gaze distance to the object. It was found that the additional features of the objects perceivability have to the vast majority of classifiers, a positive impact on the prediction accuracy. Thus, the prediction accuracy of the neural network improves by almost 15 %. Yet, it must be taken into account that the network structure used is possibly not optimally suited to classify a single value feature. However, Figure A.3 also shows that classifiers designed independently of the feature size show similar accuracy improvements.

6.2.3 Limitations

The presented model predicts the raised retinal stimulus of a scene object, taking into account the retinal perceptual abilities of the affected region. The model only depicts the preattentive stage and not the full pipeline of human visual perception. The subsequent filtering and processing of the extracted stimulus is controlled by higher cognitive processes and depends strongly on the semantic significance, performed task, or intention. In the presented straight forward experiment scenario, this attentive filtering step was successfully fitted by a small neural network. However, in more complex scenarios, this stage may need to be adapted. For instance, in driving scenarios, it is necessary to further include semantic information, such as object class or currently performed maneuver, as well as subjective attributes, such as driving experience and route knowledge. It should also be noted that a once trained classifier is likely not transferable to another application domain, and therefore needs to get fine-tuned or re-trained.

The presented experiment demonstrates the functionality of the model in a desktop scenario using a synthetic stimulus. While the results can be transferred to VR scenarios, the application on a real-world stimulus, such as in AR, has to take into account further factors, which are not reflected in the conducted experiment. In this case, the model is limited to

the color and contrast accuracy of the scene camera. Also, the depth focus of the user may vary and could differ from the displayed HUD elements.

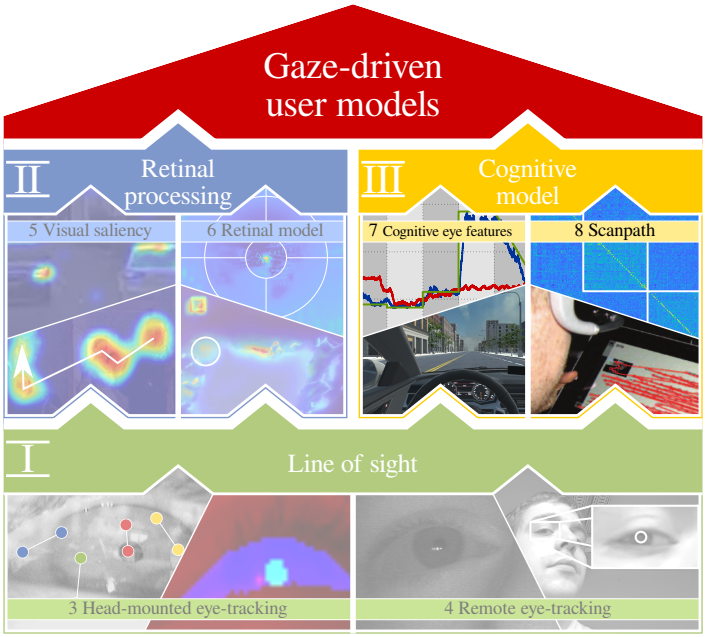
6.2.4 Conclusion

The presented model attempts to predict the user's perceivability of scene content, using the user's tracked line of sight and visual scene information. Generated features can be used to train a classifier to predict whether a certain scene object is likely to be perceived by the user or not. A proof-of-concept experiment with a Heads-Up Display demonstrated the functionality of the model as well as the valence of the extracted features. The proposed neural network increases its classification accuracy by 15 % to over 81 %, compared to gaze-only classification, about 8 % compared to Itti & Koch saliency, and about 4 % compared to a gaze driven GBVS classification.

However, retinal perception is only the first step in the user's visual perception pipeline. A complex cognitive mechanism not only controls which areas of the scene are focused, but also filters the enormous amount of retinal extracted visual information in terms of its significance and semantic relevance. It is nowadays impossible to entirely model this complex interaction of different human cognitive areas. Yet, top-level information, e.g., performed task, workload, or fatigue, may enable inferences about which extracted visual stimuli are more likely perceived consciously. Hence, the next Part III elaborates methods to predict various cognitive features from the recorded gaze signal, to refine the retinal perception model with cognitive situation awareness.

PART III

Cognitive model



Cognitive factors play an essential role in how a user perceives a scene stimulus. Depending on the mindset, performed task, workload, expertise, fatigue, social environment, and many more, the selectivity and processing of the visual input differentiate significantly. Therefore, the human brain filters the vast flood of retinal perceived and extracted stimulus (bottom-up) in a top-down paradigm, through progressive abstraction and aggregation of high-level information [307]. This selectivity of deliberate perceived visual scene content enables humans to efficiently focus their attention, i.e., on task-relevant objects in the scene [104], [247]. In a nutshell, humans efficiently filter the retinal stimulus in terms of expectation (resp. unanticipated dynamics) and semantic relevance.

However, this also implies that scene content classified as salient, nevertheless, was not consciously by the user, according to its cognitive state. In the context of driver monitoring, where visual perception models aim to contribute to model the driver's situation awareness,

it becomes therefore crucial to complement the predicted retinal perception by cognitive models to depict the full human perception capabilities.

The following two chapters introduce several methods to predict the user's cognitive conditions using an eye-tracking signal. Chapter 7 introduces low-level eye features to assess a driver's cognitive load and the performed task. However, Chapter 8 digs deeper into the task prediction by analyzing the user's visual scanpath, respective the selection of visual attention.

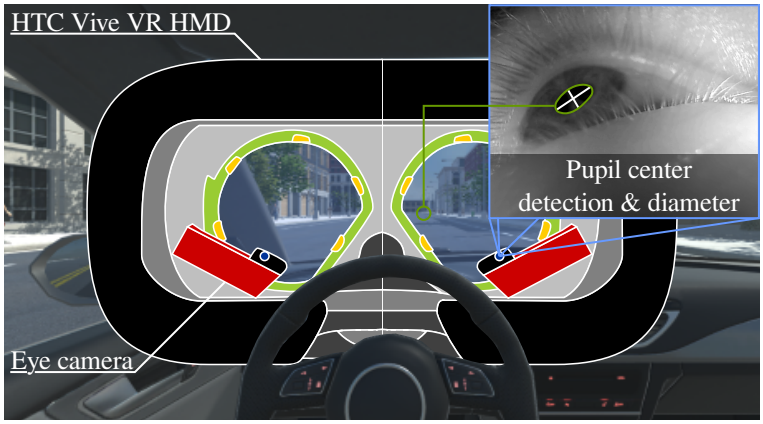
CHAPTER 7

Cognitive eye features

This chapter considers cognitive eye features as features that can be directly derived from the eye image or the pupil signal. The most prominent exponent of this feature category is pupil dilation. Pupil dilation is well established as a significant indicator of cognitive load and is extensively used in psychology research. However, the reasons for pupil dilation or contraction are versatile. The most intuitive reason for changing pupil diameter is a significant change in lighting. Thus, the pupil dilates in the dark, to catch up more light, or contracts to limit the amount of light falling onto the retina (pupillary light reflex) [308]. Yet, also in surprising, confusion, or frightening situations, the pupil dilates [309]–[311]. This stress-reflex from a time when humans had to fear natural enemies enables humans to expand their field of view temporarily and to process more incident stimulus. But, also non-hazardous situations, e.g., watching infant's face, may cause pupil dilations [312]. However, Section 7.1 assesses pupil dilation as a measurement of cognitive load in a driving scenario and attention guiding [7], [8].

Moreover, Section 7.2 introduces a novel approach for adaptive online detection of eye movements. Eye movements, i.e., fixation and saccades, enable inferences of the user's cognitive load and stress level [32]. High cognitive load leads to less frequent yet more prolonged fixations [313], whereas time pressure tended to decrease fixation duration, leading to an increasing frequency of fixations [314]. Moreover, the eye movement's distribution enables inferences about the driver's performed secondary task during automated driving [12]. In turn, the driver's secondary task reveals significant indicators to predict the cognitive mindset and workload. Both are considered as essential factors to evaluate a predicted visual perception in a top-down manner.

7.1 Realtime assessment of cognitive load



Eye-tracking enables analyzing the user's eye movements, blinks, and pupil dilatation in an online manner. Especially, the latter is known as a strong indicator of a cognitive load. In fact, the user's pupil diameter is influenced by a multitude of factors. Primarily the pupil dilates to control the light incidence to the retina. However, significant pupil dilation is also induced through increased activity of the sympathetic nervous system, e.g., during excitement, fear, joy, emotional affection, load, and other cognitive processes. Detecting such a user's cognitive states may provide valuable modalities for advanced user-centered HCI systems. In the automotive context, e.g., a driver's cognitive load may enable inferences of expertise and sovereignty w.r.t. specific driving scenarios. Moreover, in critical traffic situations, a detected driver's overload may is used to reduce traffic accidents through technical interventions. Furthermore, detecting the driver's cognitive load may enable to determine a suitable time frame to handover from conditional automated to manual driving.

Combined with a retinal perception model, such cognitive load indicators enables a detailed glimpse into the user's visual perception paired with inferences towards high-level cognitive processing capacities. Such comprehensive user models of situation awareness and responsiveness enable ingenious and extensive human-computer collaboration and open up a variety of novel and sophisticated application domains. However, most of these scenarios require cognitive load detection in a small time window. Hence, this section presents a methodology to evaluate the cognitive load in a VR driving scenario in realtime.

The following sections are supported by two prior publication from Bozkir & Geisler et al. at the Conference on Virtual Reality and 3D User Interfaces (VR) and the Symposium on Applied Perception (SAP) in 2019 [7], [8]. Section 7.1.1 presents the novel method for cognitive load assessment in a VR driving environment. Section 7.1.2 provides an extensive evaluation, followed by the limitations and conclusion in Section 7.1.3 and 7.1.4, respectively.

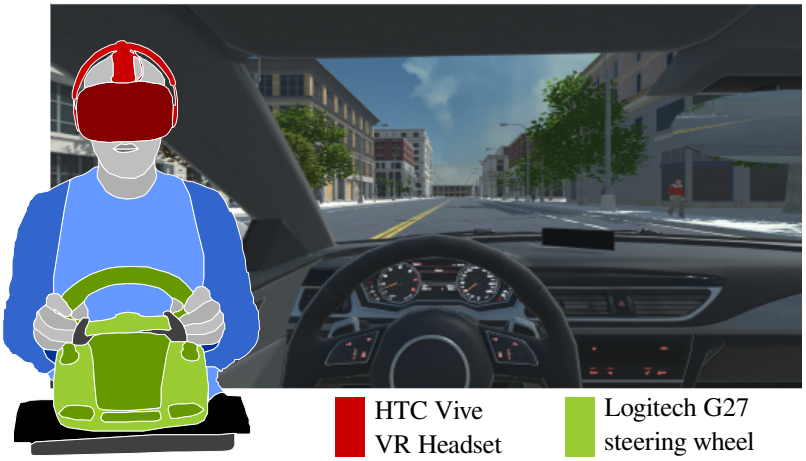


Fig. 7.1.1: The hardware setup, along with the virtual driving scene implemented with Unity3D. The scene is created using 3D models from the Urban City Pack, and City Park

7.1.1 Method

The online classification of the driver's cognitive load was developed along with a sophisticated VR driving simulation platform. The following Section 7.1.1.1 provides a comprehensive description of the driving simulation environment and the conducted cognitive load and attention guiding study. The subsequent Section 7.1.1.2 provides the actual methodology to predict the driver's cognitive load in an online manner.

7.1.1.1 VR setup and environment

In previous work, Bozkier and Geisler et al. presented a driving study to evaluate the impact of attention guidance on driving safety and cognitive load [8]. The full virtual driving scene was implemented by an HTC-Vive VR, a Logitech G27 steering wheel and pedals, Phillips headphones, and a Pupil-Labs HTC-Vive binocular eye-tracking system (see Figure 7.1.1). This setup enables a realistic driving experience, yet with full control of the simulated environment and a minimum of side effects. Furthermore, VR enables the simulation of high-risk driving scenarios, which cannot be conducted in field driving studies for safety reasons. Thus, e.g., Bozkier and Geisler et al. evaluated the effect of visual attention guidance when pedestrians suddenly cross the street [8].

The VR scenario was designed using the rendering engine Unity 3D and the model extension packs: Urban City Pack, City Park Exteriour, and Traffic Sign Sets. The pedestrians were implemented using the Modern People asset package. Moreover, the traffic objects such as vehicles, including the path generation and other helper scripts, were provided by the asset packages: Realistic Car HD02, Traffic Cars, Realistic Car Controller. Finally, the asset packages Playmaker and Simple Waypoint System provided the logic for natural, yet still controllable, pedestrian, and traffic movements. Figure 7.1.2 and 7.1.2 shows some

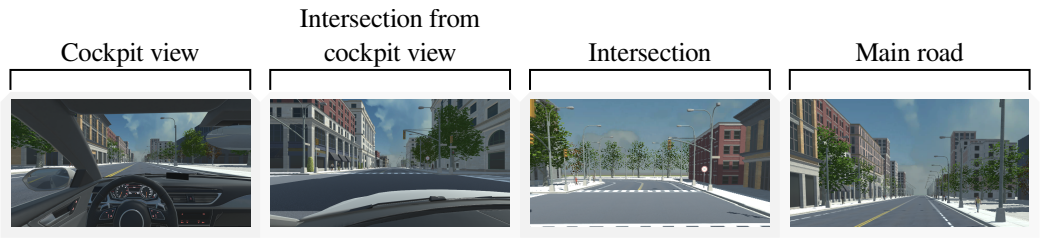


Fig. 7.1.2: Driving scene examples rendered with Unity 3D.

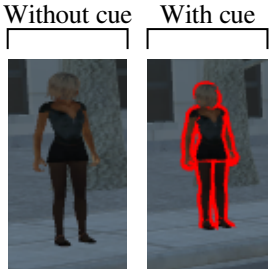


Fig. 7.1.3: Pedestrian with and without warning cues.

excerpts of the implemented simulation environment. The entire driving scenario was fully dynamic, meaning the driver was able to accelerate, brake, and steer at will. During the simulation, all important features such as the line of sight, the route, and the distance to study-relevant objects were recorded synchronously.

Experiment design The experiment comprises an acclimation and data collection phase. While the acclimation phase, the recorded data was discarded, thus the driver had time to collect experience with maneuvering the vehicle and get used to the simulation conditions. However, in the data collection phases, the driver was encountered in hazardous situations in terms of spontaneous crossing pedestrians. Therefore, two critical pedestrians on the sidewalk were generated. The pedestrians crossed the road whenever the distance to the driver fell below $d_{\text{critical}} \approx 45\,\text{m}$. Due to the simulation start position, each driver was encountering at least one critical pedestrian during the data collection phase. Half of the drivers received a warning cue at a distance of $\sim$32 m – as illustrated in Figure 7.1.3 – whenever they approached a critical pedestrian.

This parametrization kept the Time to Collision (TTC) in a range of $1.8\,\text{s} < \text{TTC} < 5\,\text{s}$ since the road's speed limit was set to $90\,\text{km}\,\text{h}^{-1}$. In advance, the driver was instructed to pay attention to speed limit signs and not to exceed it. Yet, was the speed limit exceeded an in-car maximum speed warner alerted the driver in the command panel. The pedestrian warning cues were implemented as gaze-aware, in terms of they deactivate after the driver was gazing the warning in a bounding box of 5 m for at least $\approx 0.85\,\text{s}$.

Therefore, the driver's gaze signal was projected from the 2D canvas (HMD) to the scene using ray-casting and Unity's 3D colliders. Simultaneously, the driver's pupil diameter

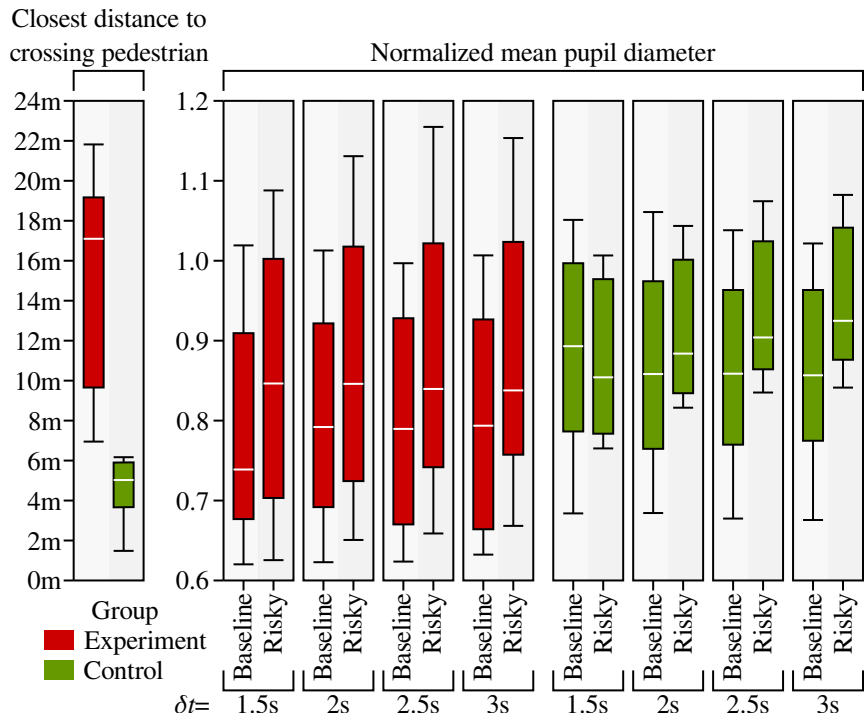


Fig. 7.1.4: The left boxplots show the closest distance to a crossing pedestrian with (control group), and without cue (experiment group). The right plots indicated the normalized pupil diameter for different time windows δt.

was recorded and post-processed using a Savitzky-Golay filter [315]. Moreover, the pupil dilation was normalized using a baseline of a 0.5 s window. Figure 7.1.4 shows the average distance to the crossing pedestrian for drivers with (experiment group) and without cues (control group). Drivers receiving warning cues have kept a significantly larger distance to the hazard situation. Moreover, Figure 7.1.4 also shows the average pupil diameter over different time windows when the driver approaches a critical pedestrian. A comprehensive evaluation of the recorded study parameters was published by Bozkir and Geisler et al. at the SAP in 2019 [8].

The simulation setup and experiment was performed on a conventional desktop computer equipped with NVIDIA Titan X GPU (12 GB memory), a 3.4 GHz Intel i7-6700 CPU, and 16 GB of memory.

7.1.1.2 Cognitive load recognition

However, the conducted study does not provide ground truth annotation for the driver's cognitive load directly. Hence, the recordings were annotated as *Low* and *High* cognitive load sequences in a post offline analyzing step. Therefore a timestamp t_{critical} separates the sequences into low and high cognitive load levels. For the experiment group, t_{critical} corresponds to the timestamp when the warning cue appeared, whereas for the control group t_{critical} corresponds to the timestamp when the critical pedestrian starts crossing the road.

Time frame	Cognitive load
$[t_{\text{critical}} - \delta t, t_{\text{critical}} + \Theta t_{\text{shift}}]$	Low
$[t_{\text{critical}}, t_{\text{critical}} + \Theta t_{\text{shift}} + \delta t]$	High

Tab. 7.1: Cognitive load annotations for time-frames

The reason for the different timestamps t_{critical} for the control and the experiment group is that the driver's cognitive load increases at that point when the critical situation became conspicuous.

In order to determine and annotate proper time frames, a T-test on the pupil diameter for each driver over the time frame $[t_{\text{critical}} - \delta t, t_{\text{critical}}]$ and $[t_{\text{critical}}, t_{\text{critical}} + \delta t]$ was performed. If the T-test reported a $p < 0.05$ for an intermediate δt, the corresponding time frames were annotated as low and high cognitive load sequences, respectively. Yet, to keep the distribution significantly different but rather close to each other, time frames with a $p < 0.01$ were skipped. However, since cognitive load also depends on some biological factors, its effect is delayed in the recognizable pupil dilation. To compensate for this delay, t_{critical} was shifted by $+\Theta t_{\text{shift}} = 0.8\,\text{s}$, as shown in Table 7.1.

Classification: However, besides the pupil diameter, the feature set was extended by further driving relevant modalities, such as driver inputs on the accelerator and brake pedals, and steering wheel. Moreover, all features were normalized by a min-max normalization. In order to predict, whether a time frame belongs to low or high cognitive load, the collected feature sets were used to train and evaluate various online capable classifiers:

Decision trees conjugate many weak classifiers (often simple comparators) as nodes in a tree structure into a powerful one [299].

Random forests ensembles a multitude of decision trees. All decision trees have developed according to a certain type of randomization during the training process. The classification is performed as the mode of single tree decisions [316].

Support Vector Machines (SVMs) classifies observations according to a trained hyperplane [305].

k-Nearest Neighbors (kNNs) classifies the observation by looking up the k nearest neighbors and counting their class according to the distance to the observation [306].

7.1.2 Evaluation

The training and evaluation took place in a one-leave-out validation scheme to ensure a driver's independent classification model. Hence, during a training session, all extracted time frames were considered, except for one driver. The subsequent evaluation was then performed against the left out driver. Thus, the training dataset does not provide any person-related information from the evaluation dataset. The training and evaluation were

Method	Acuracy	Precision	Recall	F1-Score
SVM	80.70 %	0.7671	0.8574	0.8098
Decision trees	73.44 %	0.7332	0.7005	0.7165
Random forest	74.36 %	0.7372	0.7230	0.7299
kNN				
$k = 1$	69.68 %	0.6846	0.6809	0.6828
$k = 5$	75.66 %	0.7473	0.7433	0.7453
$k = 10$	78.82 %	0.7947	0.7522	0.7729

Tab. 7.2: Results of cognitive load recognition

Method	Mean prediction duration
SVM	319 ns
Decision trees	305 ns
Random forest	5420 ns
kNN	
$k = 1$	741 ns
$k = 5$	742 ns
$k = 10$	764 ns

Tab. 7.3: Evaluation of mean prediction durations

conducted using Matlab's Classification Learner toolbox (2019b), running on moderate hardware: NVIDIA GeForce GTX 1070 mobile GPU with 8 GB memory, 2.2 GHz Intel i7-8750 CPU, and 32 GB memory.

The recorded dataset comprises a total of $1,171$ frames, whereas each driver produced in average ~ 73 annotated time frames (SD $= 12.5$). The evaluation of different feature subsets revealed that the steering wheel feature has no significant positive impact on classification accuracy. However, this likely due to the experiment scene did not require much lateral correction during the critical scenes (the road was manly straight).

Table 7.2 indicates the accuracy, precision, recall, and F1-Score of the evaluated classifiers. The highest accuracy of 80.7 % was achieved using the SVM. Moreover, it was observed that including addition driving data besides the pupil diameter, e.g., accelerator padel input, has great potential to increases the accuracy. The highest accuracies were achieved by combining pupil diameter and pedals input.

Table 7.3 indicates the runtime of the different classifiers for a single time frame classification. All tested classifiers are known to be particular runtime efficient and suitable for an online operation mode. However, considering the high accuracy, yet due to the still fast runtime performance, the SVM provides a reasonable compromise. Moreover, classifiers like kNN or random forests tend to decrease runtime performance with growing datasets.

7.1.3 Limitations

The proposed model was evaluated in a VR-based driving simulation. However, even though the full synthetic scene enables well-controlled experiment conditions, the conclusion to outdoor driving scenes is limited.

For instance, alternating sequences of incident sunlight and shading may cause a significant pupil dilatation. Even a shift of the line of sight from a shaded towards a well-illuminated scene area can be sufficient to induce a significant variation of the pupil diameter. Such challenging conditions can only be simulated virtually up to a limited extent. Even though state-of-the-art HMD provides remarkably realistic images, the color and lighting dynamic range is far from natural scenes.

Moreover, it is required to investigate to what extent the lack of lateral and steering acceleration forces impacts the driver's situation assessment. E.g., critical situations might be assessed as harmless, or vice versa. Accordingly, the cognitive load does not change as expected. However, this would not affect the classification itself, yet the ground truth annotation of low and high cognitive load sequences.

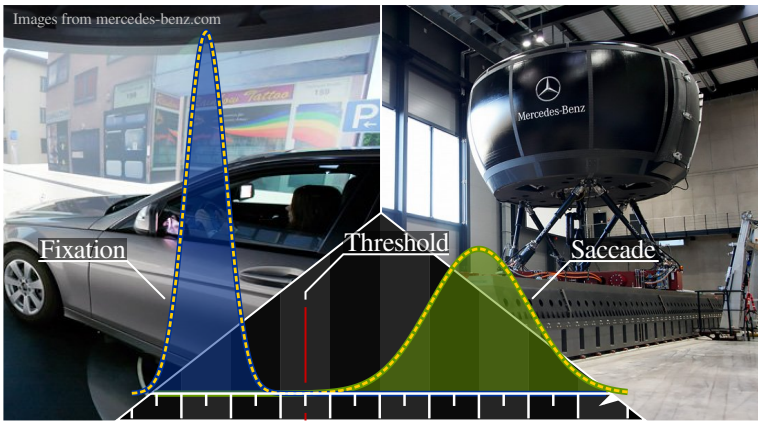
7.1.4 Conclusion

The proposed approach enables an online prediction of a driver's cognitive load in safety-critical situations. The data collection and evaluation was performed in a well controllable, yet dynamical and realistic VR driving simulation. The recorded features were evaluated in terms of predicting the driver's cognitive load in a cross-subject paradigm. The achieved accuracy of $> 80\,\%$ provides clear evidence that the drivers' pupil diameter and pedal input enables robust online prediction of cognitive load, even across different subjects.

Moreover, the time frame based metrics enable efficient implementation using sliding sampling windows. This enables a continuous reassessment of the driver's cognitive load status. The exceptional performance and lightweight classification allow embedded computing, e.g., automotive use cases. Combined with a sophisticated retinal perception model, as introduced in Chapter 5, this leads to a comprehensive estimation of the driver's scene perception and processing capacities and opens up a versatile of possible applications.

In the next step, the proposed model requires to be evaluated in a real driving experiment. However, this step raises the challenge that such critical scenarios cannot be intentionally induced as in a VR driving setup.

7.2 Adaptive eye movement classification



Eye movements are straightforward yet powerful indicators for various cognitive processes and states. Driven by conscious foveal attention shifts and reciprocal bottom-up processes to optimize the visual information gain, eye movements reveal insights into the user's visual processing.

However, the classification of eye-tracking signal samples into either fixation or saccade is far from trivial. Depending on the hardware (e.g., remote or head-mounted eye-tracking) and the underlying software stack (see Section 2.1), a sample may represent completely different measurements. Yet, remote eye-tracking systems may report the line of sight in a 3D scene coordinate system. Whether a sample relates to fixation or saccade usually hinges on the extent of deviation from the previous sample in terms of its spatial interpretation. Hence, a popular approach is to calculate the distance on the first temporal derivative over the eye-tracking signal: the eye movement velocity v_i. Thus, differently spatially dimensioned eye-tracking signals (2D or 3D) are reduced to a single value per sample. However, different velocity units, e.g., px s^{-1} or $^\circ$ s^{-1}, make generalized eye movement classification challenging. Furthermore, various influences, such as signal noise and sampling rates, may limit the classification in several aspects. For instance, saccades pronouncing a lower average velocity in the timespan between v_i and v_{i-1} than the noise level of v are not distinguished from fixations.

At the same time, the interpretation of a saccade and fixation often depends on the assessed context. For instance, in a driving scenario, the evaluation often focuses on large saccades between different objects. Small eye movements within an object (e.g., reading letters on the license plate), or even microsaccades are usually reported as fixation. However, in a reading context, the evaluation may focus on small saccades between letters, words, and lines. Even microsaccades may of interest to assess cognitive load. In such a scenario, the classifier must be as sensitive as possible, yet still not susceptible to noise.

A successful and wide-spread approach is the use of GMMs [317]–[319]. These models optimize an assemble of two or three (considering smooth pursuits as an extra class of eye movements) distributions on the observed eye movement velocities v_i. Assuming that fixation and saccade velocities follow two distinct distribution, such models provide the optimal split of fixations and saccades. Moreover, GMMs work independently of any velocity units and are robust against noise. Furthermore, such models may even depict different levels of sensitivities related to the performed task, as, e.g., in reading small eye movement dominate and thus more likely modeled as saccades, whereas in driving large saccades between distinct scene objects shift the classification threshold towards a less sensitive classification.

However, state-of-the-art GMM approaches, such as the Bayesian Mixture Model (BMM) from Kasneci (former Tafaj) et al., suffer on long term studies, comprising multiple different tasks, where the underlying distributions may shift dynamically [317].

The following sections are supported by a prior publication from Braunagel & Geisler et al. at the conference on Eye-tracking Research & Applications (ETRA) in 2016 [12]. Section 7.2.1 introduces a novel online adaption of GMM to classify eye movement velocities into fixation and saccades. Moreover, using sliding estimators, the presented approach provides minimal hardware requirements in terms of memory and processing capacities, making it excellent suitable for online signal processing in embedded systems. Section 7.2.2 provides a comprehensive evaluation of the proposed method, including an extensive driving simulation study and benchmarks to BMM and I-DT. The final Sections 7.2.3 and 7.2.4 state the limitations of the presented approach and the final remarks.

7.2.1 Method

This section introduces Moving Estimation Classification (MERCY), a novel approach for continuous online estimation of eye movements BMM classifier. Figure 7.2.1 shows the fundamental procedure of MERCY divided into three iterative steps: Step ① eye movement classification, Step ② model estimation, and Step ③ updating of the classification model. These steps are iteratively performed with each new sample $||v_i||$ at time i (assuming constant sampling rate), and the estimated BMM parameters $\Theta_{f\ i-1}$ and $\Theta_{s\ i-1}$ of the previous iteration $i-1$.

The measure $||v_i||$ indicates the eye movement's absolute velocity at time i. In a straightforward implementation, $||v_i||$ is approximated by a mere discrete differentiation of an eye-tracking pupil signal. However, a more accurate and noise-robust approach is to process the first derivative of a Savitzky-Golay filter at time i (analog to Section 7.1) [315]. The distribution parameters Θ_f and Θ_s describe the Gaussian likelihood models for an observed fixation or saccade, respectively.

Step ① Classification MERCY classifies a measured eye movement $||v_i||$ as a regular BMM by assessing its likelihood w.r.t. the two mixed models $\Theta_{f\ i-1}$ and $\Theta_{s\ i-1}$. Hence, the intersection δ of the two models is considered as a velocity threshold, as illustrated in Figure 7.2.2. Eye movements with a velocity lower than δ (left-sided) are classified as fixations, whereas eye movements faster than δ (right-sided) are classified as saccades.

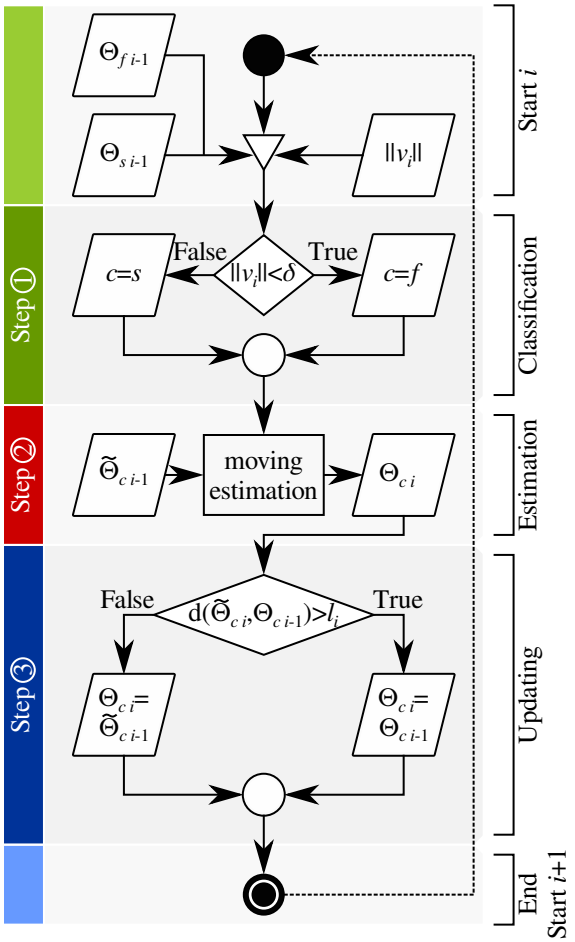
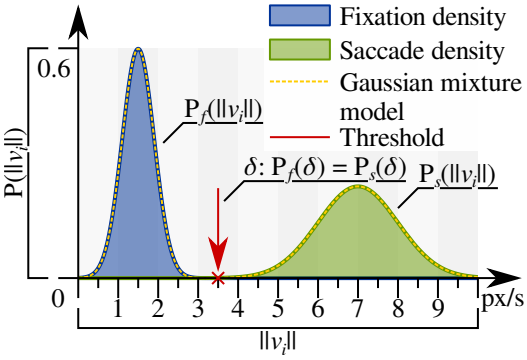


Fig. 7.2.1: The architecture of the MERCY algorithm.

Fig. 7.2.2: The three probability density functions $P_f(||v_i||)$ (blue), $P_s(||v_i||)$ (green), and $P(||v_i||)$ (dashed yellow) of the eye movement GMM.



After the classification, MERCY incorporated the measured eye movement to adapt the respective w.r.t. the new sample $||v_i||$.

Step ② Estimation MERCY continuously estimates the two parameter sets Θ_f and Θ_s by means of the sampling mean, variance, and a priori of the observed eye movements $||v_i||$. The empirical estimation of means, variances, and mixture components can be reformulated into a recursive form (moving sample estimators). However, these estimators tend to converge as the number of samples increases; thus, new data samples are incorporated with decreasing weights. Hence, MERCY implements an additional weighting factor ω, as a moving, exponentially decreasing, window size. Thus, choosing a small ω emphasizes new samples and leads to a very dynamic estimation, yet increases the impact of outliners on the estimation. Vice versa, choosing a large ω leads to an idle response and adapts only slowly to changing conditions.

Assuming that the recorded eye movement velocities $v_1, v_2, \ldots$ are realizations of the random variables $\mathcal{V}$ generated by an independent and identically distributed process, the recursive equation of the weighted sample mean and variance is defined as

$$\mu_{k\ n+1} = \frac{\omega \cdot \mu_{k\ n} + v_{n+1}}{\omega + 1}, \tag{7.1}$$

and

$$\beta_{k\ n+1} = \frac{(\omega - 1) \cdot \beta_{k\ n} + (V_{n+1} - E\left[V_{n+1}\right])^2}{\omega}, \tag{7.2}$$

respectively. Note, the variance $\beta_{k\ n+1}$ estimation depends on the mean estimation $\mu_{k\ n+1}$ of the same iteration.

The mixture components π_f and π_s describe the apriori to observe a fixation or saccades sample, respectively. Both are estimated as a ω weighted counter as follows:

$$\pi_{k\ n+1} = \frac{\omega \cdot \pi_{k\ n} + 1}{\omega + 1}. \tag{7.3}$$

In contrast to the sample mean and sample estimation, the two parameters π_f and π_s, are updated in each iteration of the algorithm (since $\pi_f = 1 - \pi_s$).

Step ③ Update As illustrated in Figure 7.2.1, MERCY uses two pairs of parameter sets, Θ_k and $\tilde{\Theta}_k$. While the parameter sets Θ_k provides the classification model parameters, $\tilde{\Theta}_k$ implements the continuously adapted sample parameters. However, whenever these two parameter sets diverge more than a pre-defined threshold l, $\tilde{\Theta}_k$ is updated by $\tilde{\Theta}_k$. This procedure prevents the classifier to adapt too fast to outliers or monotonous sequences (e.g., long fixations). Yet, at the same time, the classification model is able to adapt to significant changes in the eye movements distribution. However, setting $l = 0$ leads to an update of $\tilde{\Theta}_k$ on each new data sample, which is, in general, the proper approach. Nevertheless, this separation into two parameter sets was considered as possible additional analysis of the task-

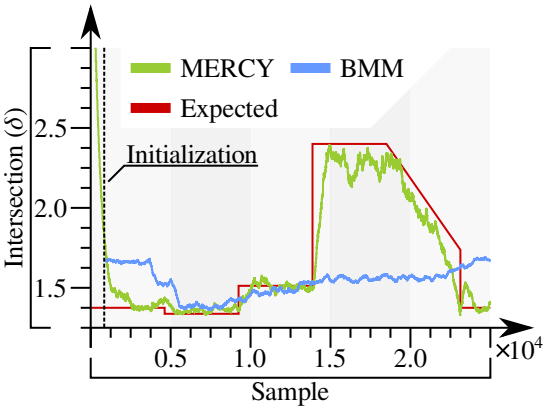


Fig. 7.2.3: Benchmark of MERCY (green) and BMM (blue) w.r.t. an artificially generated threshold (black). After an initialization phase, both MERCY and the BMM adapt to the given threshold. Yet, at $\sim$15000 samples, BMM is overfitted and not able to adapt appropriately to threshold changes anymore.

and interindividual differences. Moreover, the two model approach enables a more robust parameterization against overfitting in monotonous sequences (see Section 7.2.3)

Since all three steps provide a constant complexity $\mathcal{O}(1)$, MERCY fulfills the requirements for real-time applications. Moreover, except for the distribution parameters, MERCY operates directly on the data stream without any memory significant memory consumption.

Figure 7.2.3 shows the adaption of MERCY to a sequence of artificial eye movements compared to a classical BMM. Although MERCY was unfavorably initialized with an offset of 0.5 for means and variance, the proposed algorithm adapts fast as the BMM to the artificial model. Moreover, at $\sim 15,000$ samples, MERCY plays out its adaption mechanism of Step (3) and outperforms the BMM significantly. The BMM incorporates all observed samples with the same weight. Hence, it increasingly converges and is not able to adapt to changes in the eye movements distribution anymore.

The next section provides a comprehensive evaluation using real-world eye movements from a large-scale driving simulation experiment.

7.2.2 Evaluation

The evaluation splits into two parts: 1) an extensive driving simulation study is presented in Section 7.2.2.1, on where realistic data in terms of natural eye-movements while driving and performing secondary tasks where collected. 2) Section 7.2.2.2 provides the results in terms of a statistical analysis of the eye movement's distribution on different tasks and how accurate MERCY adapts the classification threshold in an online operation mode.

7.2.2.1 Experiment

The evaluation data was collected on an extensive conditional automated driving study at Mercedes-Benz's moving based driving simulator. The simulator provides a 360° scene projection with realistic traffic, wind, and engine sound effects. Moreover, the moving platform provides an authentic simulation of realistic physical forces in all 6 degrees of freedom through moving a hexapod mounted on a 12 m long sled. All participants drove a typical german highway for $\sim$35 min at $\sim$120 km h^{-1} in a realistic Mercedes-Benz W212 E-Class cabin.

Algorithm	Recall	Precision	F1 score
I-DT	0.73	0.66	0.69
BMM	0.86	0.67	0.75
MERCY	0.91	0.75	0.82

Tab. 7.4: Classification results of I-DT, BMM, and the proposed MERCY.

The experiment starts with an acclimatization phase to introduce the participants to the driving simulator, the automated and manual driving scenario, and to avoid training effects. While the acclimatization phase, the participants had no secondary task during automated driving. Moreover, four take-over situations were implemented to familiarize the driver with the handover from automated driving to manual driving and vice versa.

While the experiment phase, a touch screen mounted in the front of the center console provided secondary tasks, which the driver was instructed to perform during automated driving. The secondary tasks included watching a movie (Partly Cloudy [293]), reading news, writing an email, listening to music, and idle, i.e., the driver was asked not to perform any task. Moreover, the manual driving sequences were considered as an additional task.

The drivers' eye movements were recorded using a Dikablis head-mounted eye-tracking with a sampling rate of 25 Hz. In total 85 drivers were recorded in this study, divided into an experimental group of 74, performing secondary tasks, and a control group of 11 test subjects, performing only the idle task. The cabin was equipped with four video cameras, mounted at different angles for permanent observation of the driver's face, paddles, steering wheel area, and the touch screen.

An extensive elaboration of the experiment setting is given in the related publications of Braunagel and Geisler et al. [11], [12]

7.2.2.2 Results

The evaluation was performed on 74 drivers as 6 drivers experienced critical simulation faults such as traffic freeze. Moreover, 5 drivers have been discarded because of an eye-tracking signal loss. In total, the eye movement data set comprises 35.5 h of recorded eye-tracking data separated into 1.5 h of manual and 34 h of conditionally automated driving respective task performing.

In order to evaluate the overall accuracy of MERCY compared to I-DT and BMM, the eye movements of 6 drivers were manually labeled in terms of fixations and saccades. The resulting ground truth dataset comprises $6,623$ fixation and $1,384$ saccade samples. Table 7.4 shows the achieved accuracy, precision, and F1-score of the respective algorithms. The I-DT threshold was set to 100 ms and 15 px in terms of the eye camera resolution. MERCY achieved the highest scores in all three assessed metrics, followed by the BMM.

However, to gain a deeper glimpse into the dataset and the performance advantage of MERCY, the dataset was investigated w.r.t. its underlying eye movements $||v_i||$ distribution. Therefore, the whole dataset was partitioned to idle sequences, where the driver did not perform any task, and task sequences, where the driver either was manually driving or run

	Task					Idle				
	Mean	Med.	Var.	Min.	Max.	Mean	Med.	Var.	Min.	Max.
μ_f	0.73	0.69	0.08	0.26	1.96	0.53	0.50	0.03	0.23	1.46
μ_s	55.4	52.4	337.9	21.7	89.0	50.4	46.4	262.7	21.5	80.6
β_f	2.26	1.96	2.51	0.55	9.73	1.32	1.15	0.65	0.49	6.00
β_s $(\times 10^3)$	3.60	3.73	242	1.76	4	3.62	3.76	180	1.83	4
π_f	0.91	0.92	< 0.1	0.73	0.98	0.91	0.92	< 0.1	0.71	0.97
π_s	0.09	0.08	< 0.1	0.02	0.27	0.09	0.08	< 0.1	0.03	0.29

Tab. 7.5: Statistical values of the estimated GMM for Task and Idle sequences.

of the secondary tasks (watching video, reading, writing an email, listen to music). On both partitions, an GMM was fitted using EM. Table 7.5 shows that there are substantial differences between the distributions of idle and task sequences. The strong performance of MERCY can be explained by its ability to adapt these variations in the underlying distribution, whereas GMM and I-DT perform a global approximation of the classification threshold δ and are not sensitive to temporal distributional shifts.

Figure 7.2.4 shows the distribution of the threshold determined by Mercy, divided into idle and task, whereas the task is further breakdown into the distinct task types watching, i.e., video, reading, writing mail, driving, and listening to music. MERCY's approximated threshold shows a distinct variation w.r.t. the assessed type of (non-)task sequence. Comparing the distribution of MERCY's estimated thresholds on idle and task sequences reveal a noticeable difference between the thresholds in idle and task sequences analogous to the EM-based results in Table 7.5. However, inspecting the distinct tasks, MERCY even shows clearly different threshold approximations. Hence, MERCY also enables a fuzzy classification of different performed tasks just by analyzing the distribution of saccade and fixation velocities $||v_i||$ over time.

7.2.3 Limitations

The dynamic behavior and adaptation of MERCY can also lead to overfitting in some cases. Especially in the case of an extremely prolonged fixation (e.g., caused by a faulty pupil detection), MERCY estimates the a priori π_f that high that the saccade distribution ultimately degenerates. Subsequent samples are most likely classified as fixations. As a result, the fixation distribution increasingly approximates the total distribution of the velocities $||v_i||$. Although not observed in practice, this may also happen vice versa.

On the one hand, this can be counteracted by a suitable choice of the window size ω and the model adaption threshold l. However, the presented experiment in Section 7.2.2.2 required to implement some *guards*, in order to prevent MERCY to overfit fixations on longer sequences of signal lost. Such a *guard* might ensure that the fixation a priori stays in a reasonable range or resets the whole system if overfitting was detected.

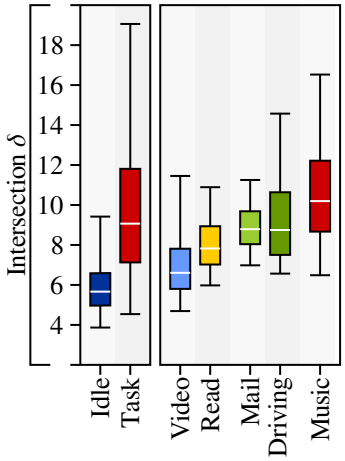


Fig. 7.2.4: Distribution of the averaged estimated intersection point δ of MERCY while performing secondary tasks versus being idle. The boxplots show the inner 95 % of the data, excluding the outer 2.5 % of the data as outliers.

7.2.4 Conclusion

This section introduced MERCY, an online GMM estimator for fixation and saccade classification. Based on sliding estimators, MERCY is able to adapt the predicted distributions of eye movement velocities of fixations and saccades in an online manner. Moreover, MERCY has shallow requirements in terms of memory and processing capacity, and thus optimally suited for high frequency and embedded applications.

An extensive evaluation on a large scale driving simulator experiment showed that MERCY successfully adapts the classification threshold to different tasks. The results in Figure 7.2.4 even indicate that MERCY can be used to fuzzy classify different tasks in the performed experiment. However, the next Section 8.1 introduces a more powerful scanpath method to classify the performed task, based on MERCY's eye movement classification.

CHAPTER 8

Scanpath

Visual scanpaths encode the user's visual attention shifts over time. The spatio-temporal elaboration of fixations and saccades reveal thrilling glimpses into the selection of fixated scene objects and the user's visual exploration strategy. This profound insight enables various inferences about the user's cognitive states, e.g., cognitive load, expertise, performed task [11], [197], [208].

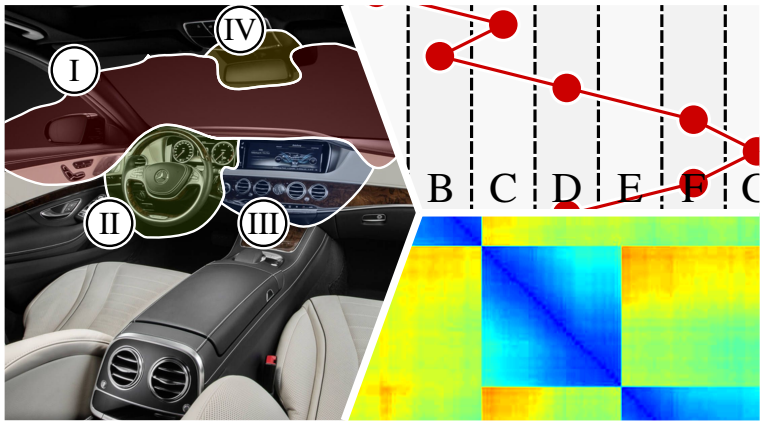
However, analyzing the scanpath is challenging and complex. Depending on the scanpath encoding, A multitude of potential features can be extracted and evaluated, ranging from classical gaze patterns to changes in fixation and saccade frequencies. A widespread approach is to bin the spatial distribution of fixations in a temporal window. The binning is might implemented by spatial quantiles or, if available, scene AOIs [11], [197], [208]. However, the former directly works on the pupil signal, whereas the latter requires an accurate mapping of the line of sight into the scene coordinate system.

The quantification attributes each fixation point to a symbol. Repetitive or supplementary symbols may identify longer fixations. Thus, two distinct subsequent symbols in the scanpath string indicate a transition of the gaze between two AOIs or bins, respectively. However, transitions may also directly encoded as distinct symbols. Therefore, each tuple of distinct symbols is replaced by a new symbol. Alternatively, the first derivative of the gaze signal is may used. Such a scanpath string does not encode the sequence of fixated scene areas, but rather the sequence of performed saccades. These transitions are known to be particularly significant in terms of the performed task.

The major challenge is to classify these scanpath strings with respect to a specific attribute, e.g., performed task, user's expertise, fatigue. Traditionally, the classification base on string (editing-) distance algorithms, such as the Levenstein distance [202]. However, these approaches quickly reach their limits, especially on long scanpaths. Modern approaches therefore match substrings which achieves significant better runtimes as well as classification accuracy [1], [197], [208].

The following Section 8.1 introduces an extensive driving simulation study, where the driver's scanpath was analyzed to predict the performed secondary task during automated driving. However, Section 8.2 addresses the runtime of scanpath analysis and introduces a novel method based on MinHash for large-scale scanpath analysis.

8.1 Online recognition of driver-activity



THE next level of automated driving enables the driver to hand over the responsibility of the vehicle to the automated driving systems. Such a system may control the full driving task, yet require the human driver as a fallback if the driving task exceeds the system's limitations. However, by giving up the actual driving task, the driver is out of the loop and may perform non-driving related tasks (in the following referred to as secondary tasks). Yet, for save and convenience take over procedures from automated driving to manual driving, the automated driving system requires to predict the driver's takeover readiness. As a key feature, detecting the driver's performed secondary task reveals insights to the driver's cognitive load and thus top-down preference of focused and perceived scene objects.

The previous Section 7.2 outlined that eye features may contribute to classifying the driver's performed secondary task. However, this section digs deeper into the prediction of performed tasks by analyzing the driver's eye movements, respectively, visual attention shifts over time: The visual scanpath. Detection characteristic patterns in the driver's scanpath, e.g., saw-tooth like horizontal eye movements repetitions on reading, enable inferences about the performed task, cognitive load, and may provides top-down parameters to predict the driver's visual perception and situation awareness.

The following sections are supported by a prior publication from Braunagel & Geisler et al. at the conference on Intelligent Transportation Systems Magazine (ITSM) in 2017 [11]. Section 8.1.1 combines the previously introduced eye movement classification MERCY [12] with the state-of-the-art scanpath classification approach SubsMatch [197], [208], to predict the driver's performed secondary task in a conditional driving setup and online operation mode. The evaluation, based on an extensive driving simulation study (same as MERCY), show an astonishing predictive accuracy of the driver's secondary task in Section 8.1.2 and supports the evidence of the power of scanpath analysis. The final Sections 8.1.3 and 8.1.4 state the limitations and final remarks, respectively.

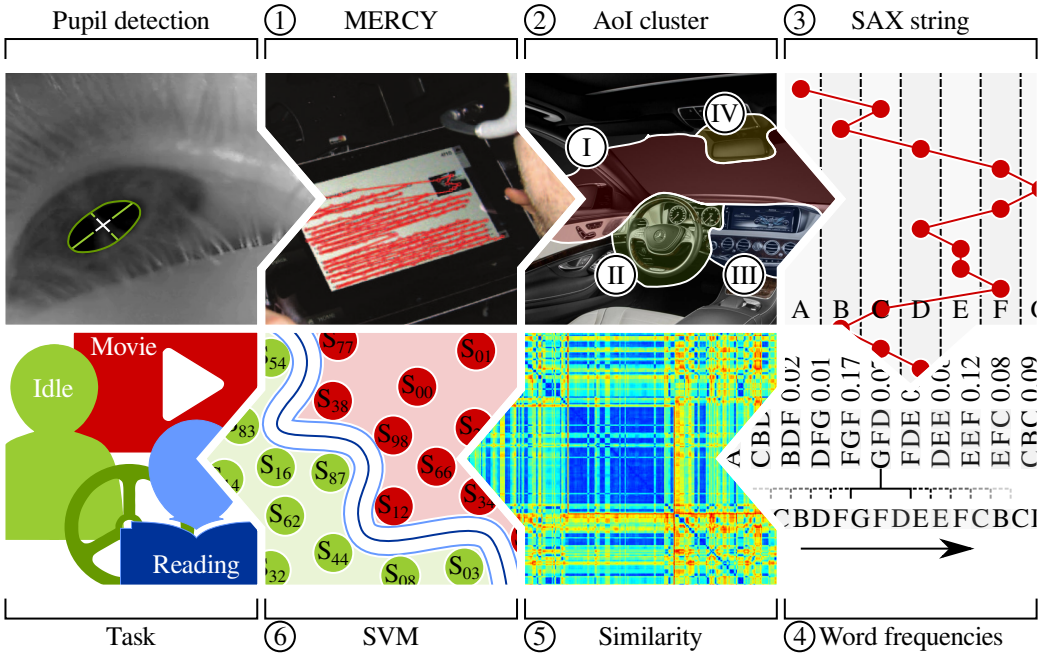


Fig. 8.1.1: Workflow of the proposed method for driver's secondary task classification. In step (1), the recorded eye movements classified into fixation and saccades using the previously introduced MERCY algorithm (Section 7.2). In step (2), the classified fixations are clustered into several AOIs. Step (3) to (5) perform a SubsMatch scanpath analysis and similarity processing. Step (6) classifies the scanpath similarity vector to the driver's performed secondary task.

8.1.1 Method

The proposed method is subdivided into six major steps as illustrated in Figure 8.1.1: 1st) Driver's eye movement classification using MERCY. This step is analog to Section 7.2 and is not further elaborated here. 2nd) Fixation cluster to identify the primary AOIs. 3rd) Scanpath discretization and encoding (SAX-strings). 4th) Word scanning and frequency counting. 5th) Scanpath similarity calculation. 6th) SVM-based task classification. The following elaborate the steps (2)-(6) in detail.

Step (2) AoI cluster After classifying the eye movements into fixations and saccades using MERCY, the spatial fixation locations are clustered using a multivariate GMM. The GMM is trained analog to the online approach of MERCY, yet in a dedicated training phase:

$$\mu_i = \begin{cases} \mu_{i-1} + \frac{\Theta_i - \mu_{i-1}}{\omega} & \text{training phase} \\ \mu_{i-1} & \text{application phase} \end{cases}, \tag{8.1}$$

and

$$\Sigma_i = \begin{cases} \begin{pmatrix} \hat{\Sigma}_i(\theta,\theta) & \hat{\Sigma}_i(\theta,\phi) \\ \hat{\Sigma}_i(\phi,\theta) & \hat{\Sigma}_i(\phi,\phi) \end{pmatrix} & \text{training phase} \\ \Sigma_{i-1} & \text{application phase} \end{cases}, \tag{8.2}$$

where Θ_i describes the tuple of the gaze's pitch and yaw angle, θ and ϕ, respectively, at time i. Analogously, Σ_i describes the multivariate covariance, whereas its components $\hat{\Sigma}_i(\theta,\theta), \ldots, \hat{\Sigma}_i(\phi,\phi)$ are the recursively estimated weighted covariance of the angle tuples $(\theta,\theta), \ldots, (\phi,\phi)$.

This divides the fixation points into a preset number of clusters. The initial parameterization of $\mu_{i=0}$ and $\Sigma_{i=0}$ control, to a certain extent, the growth and formation of clusters. However, depending on the available computing capacities and runtime requirements, a classic EM approach may also is sufficient. This dynamic clustering approach enables a driver individualized AoI detection without projection of the gaze signal into vehicle coordinates. In particular, for head-mounted eye-trackers, such a projection step may imply high technical requirements and a sophisticated software stack, e.g., as introduced Section 2.1. Moreover, static clustering implies additional challenges incorporating dynamic Aois, such as handheld devices.

However, the clustering enables a pre segmentation of fixation points into task prediction (non-)relevant. Thus, fixations falling into the windshield or mirrors (eyes on-road) are automatically classified as non-secondary task-related, and not considered in the subsequent processing steps ③ to ⑥.

Step ③ SAX-string The Symbolic Aggregate Approximation (SAX), decomposes the multivariate scanpath in terms of a sampled sequence of spatial fixation points (θ,ϕ) into symbolic string representation Σ_{SAX}. This transformation is the first step of SubsMatch, a method presented by Kübler et al. in 2014 to calculate string, respective scanpath distances [197], [208]. Therefore, the fixation points' spatial arrangement (in this case, the angular coordinates (θ,ϕ)) is reduced to symbols using a discretization rule, e.g., horizontal binning, as shown in Figure 8.1.1.

However, the crux is the definition of a descriptive discretization rule. The performance of the later classification crucial depends on the thus encoded features. Considering the example given in Figure 8.1.1 step ③, horizontal eye movements encoded as symbols `A-I`, yet vertical attention shifts are discarded. This discretization pattern is particularly expressive for, e.g., reading, since the eyes move horizontally on a line from left to right followed by a large saccade back to the next line beginning (in western typography). However, the extraction pattern will likely fail in predicting driver's mirror check sequences, e.g., left mirror → center mirror → left mirror, as the important information of the vertical offset of the center mirror to the side mirrors is not encoded. Hence the extracted scanpath pattern is not discriminating to a fixation sequence such as left mirror → control panel → left mirror.

Step ④ Word frequencies This step implements the core of the SubsMatch algorithm. A sliding window continuously processes the created SAX-string by extracting words and counting their frequencies, as illustrated in Figure 8.1.1. Therefore, the SAX-string is

preprocessed in terms of removing sequences of single symbol repetitions. Such symbol repetitions occur in particular whenever the Scanpath does not exhibit any major changes, and thus successively is reduced to the same symbol. However, this is no exceptional case. E.g., considering a 25 Hz eye-tracking system and a fixation duration of 1 s led to a symbol repetition of 25 times in the SAX-string. However, using a sliding window of size 4, this led to a high frequency of words providing only one symbol.

Although such words may indirectly encode fixation durations, their informative value is low compared to words comprising distinct symbols, due they do not encode any transition between fixations. However, such transitions are known as particularly descriptive, as they may reflect eye movement patterns that can often be attributed to a particular task, e.g., sawtooth-like eye movements in reading.

The repetition free scanpath j is described in terms of the included word frequencies as S_j. Each element in vector S_j represents provides the relative frequency of the corresponding word. Note that the number of distinct symbols in the SAX-string has an exponential influence on the length of the vector S_j.

Step (5) Scanpath similarity In a straight forward approach, the distance of two scanpath frequency vectors S_n and S_m is processed as the euclidean distance:
$d\left(S_n, S_m\right) = \sqrt{\sum_k \left(S_n\left(k\right) - S_m\left(k\right)\right)^2}$.
The resulting distance matrix over all scanpath pairs provides a deeper glimpse into the scanpath feature space, as shown in Figure 8.1.2. The raw distance matrix over 42 drivers shows that although there is a cluster formation in the respective driving tasks, it is not particularly pronounced. This is due to the fact that many words are non-discriminatory and frequently occur in each of the tasks, or are extremely rare and sometimes only occur sporadically. It is therefore recommended to perform a feature selection in advance to reduce the feature space to a small, yet significant word subset.

The Fast Correlation Based Filter (FCBF) estimates the feature significant based on its entropy [320]:

$$\mathrm{IG}\left(X\middle|Y\right) = H\left(X\right) - H\left(X\middle|Y\right), \tag{8.3}$$

where $X = \{x_0, \ldots, x_n\}$ provides the observed frequencies of a word x_i over all Scanpaths $S_0, \ldots, S_n$. Y provides the observed class, i.e., $Y = \{\mathrm{Video}, \mathrm{Read}, \mathrm{Idle}\}$. $H\left(X\right)$ is the entropy of the word X defined as follow:

$$H\left(X\right) = -\sum_i P\left(x_i\right) \cdot \log_2\left(P\left(x_i\right)\right), \tag{8.4}$$

where $P\left(x_i\right)$ is the likelihood of the word x_i in scanpath S_i (basically the normalized frequency). $H\left(X\middle|Y\right)$ describes the entropy of the word X under the condition to observe a corresponding class Y and is defined as:

$$H\left(X\middle|Y\right) = -\sum_j P\left(y_j\right) \cdot \sum_i P\left(x_i\middle|y_j\right) \cdot \log_2\left(P\left(x_i\middle|y_j\right)\right), \tag{8.5}$$

where $P\left(y_j\right)$ is the apriori of the class $y_i \in Y$ in scanpath S_i. Analogously, $P\left(x_i\middle|y_j\right)$

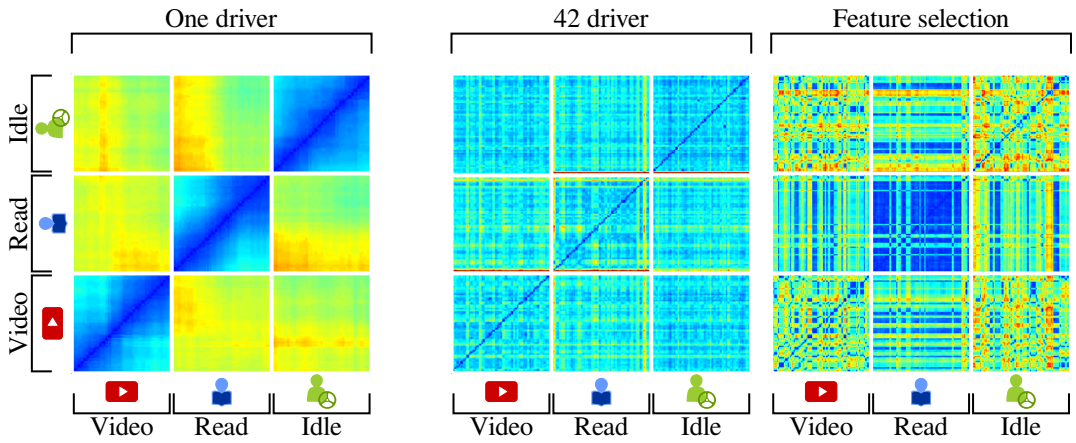


Fig. 8.1.2: Different instances of similarity matrices. Each row and column corresponds to a scanpath respective driver. The more red matrix value, the larger is the distance (resp., the lower the similarity) of the two scanpaths. The first matrix shows the similarity of subsets of the scanpath on a driver to its self. The clusters are clearly pronounced according to the performed secondary task. However, the second matrix shows the scanpath similarity between different drivers. Although the clusters are recognizable, they are much less pronounced. The last matrix shows the scanpath similarity after the feature selection. The reduction of the scanpath to significant gaze patterns (words) yields distinctive clusters over several drivers.

describes the conditional likelihood to observe the word x_i given the class y_i.

Figure 8.1.2 shows significant pronounced clusters after the entropy-based feature selection.

Step ⑥ Classification The final step consists of training an SVM to classify the scanpaths after the feature selection to the driver's performed secondary task, i.e., reading, watching a movie, or idle (do nothing). Thereby two distinct approaches may be followed: first, training and application of the SVM directly to the scanpath word frequency vector S_i. This approach is suggested in the revised version of SubsMatch (2.0) [208]. However, alternatively, the SVM can also be applied to the distance vector D_{S_i} to other scanpaths (as calculated in step ⑤):

$$D_{S_i} = \left(d\left(S_i, S_0\right), \ldots, d\left(S_i, S_n\right)\right). \tag{8.6}$$

Both approaches provide advantages and should be evaluated for the respective application case. The first approach enables direct use of the word frequency vector. The trained SVM model is, therefore, applicable without further preprocessing, fast, and resource-efficient. However, the second approach promises a better accuracy in some applications (see also the following Section 8.2), but requires the retention of the scanpath word frequencies $S_0, \ldots, S_n$, as well as a $n\times$ distance calculation.

The next section evaluates the proposed algorithm for scanpath classification on a large-scale driving simulation study. However, since the method was developed to perform online in a car embedded environment, the evaluation focuses on the original classification approach,

as suggested by SubsMatch 2.0 [208]. Yet, Section 8.2 provides an evaluation using the scanpath distance vector.

8.1.2 Evaluation

The evaluation was performed at the same experimental data as the previously introduced eye movement classification approach MERCY in Section 7.2. The dataset comprises 82 drivers (46 male / 36 female, avg. 38 years old), whereas 14 driver belongs to a control group and did not perform any task. The analyzed secondary task sequences provide about 9 h of reading, 7 h of watching a video, and 3 h of idle. Moreover, the secondary tasks reading and watching video further splits into 50 %, where the driver performed the task on a handheld tablet, and 50 % on an integrated tablet, respectively. Furthermore, the sequences were oversampled into subsequences of 5 s to 90 s with a step size of 2 s. Hence, the final evaluation dataset provides about 22 h of reading, about 17 h of watching a video, and about 6.5 h of idle.

All evaluations were performed in a leave-one-out cross-validation paradigm, where the data of one driver was separated as a test dataset, and the remaining data was used to train the model. In addition to the approach elaborated in Section 8.1.1 (in the following referred to as SubsMatch), the evaluation also includes the approach of Baunagel et al. from 2015 (in the following referred to as Baunagel et al.) to provide a profound ranking of the achieved results [321]. Braunagel et al. proposed a feature set, extracted of eye and head movements in a fixed time window, e.g., average head rotation, the variance of head rotation, mean eye movements, the variance of eye movements, and many more. In total, Braunagel et al. extract 90 distinct features, grouped in 62 saccade, 5 fixation, 3 blink, and 20 wordbook related features. The resulting feature vector is classified using an SVM w.r.t. the performed secondary task.

Classification Figure 8.1.3 shows the accuracy and F1-Score of both approaches, for handheld vs. hands-free and secondary task prediction on different sampling window sizes. The proposed approach outperforms the state-of-the-art from about 2 % up to 8 % in the prediction of whether the driver performed the secondary activity on a handheld or integrated tablet in both metrics. Moreover, predicting whether the driver performs a secondary task, the evaluation shows an improvement of about 3 % up to 17 %.

However, Figure 8.1.4 provides a deeper glimpse into the predictiveness of the proposed model w.r.t. the performed secondary task. The proposed model provides an average higher recall in the secondary task prediction, yet shows significant improvements in the precision on driver's idle sequences, where no task was performed.

Feature value Figure 8.1.5 assesses the evaluation of the selected features in step (5) by FCBF. The first plot shows the range of the number of words in the performed leave-one-out cross-validation. Except for the sequence length of 10 s, the number of selected features ranges about 36 and 71, with an average median about 52. Thus, the reduced features space by FCBF is significantly smaller as the 90 features proposed by Braunagel et al. Moreover,

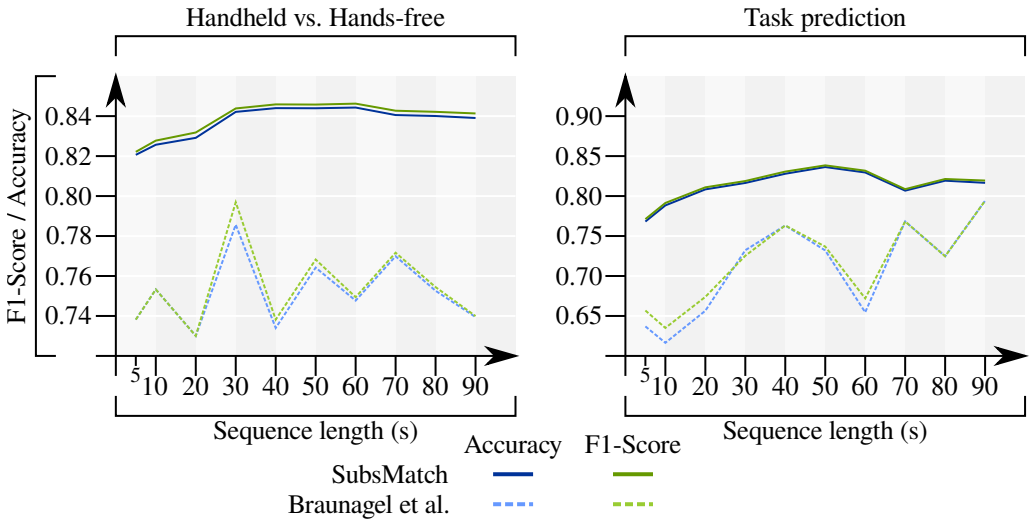


Fig. 8.1.3: The left plot shows the F1-Score and accuracy od the classification of whether the secondary task was performed on an integrated (hands-free) or handheld tablet. The right plot shows the F1-Score and accuracy in the classification of whether a secondary task was performed or the driver was in idle.

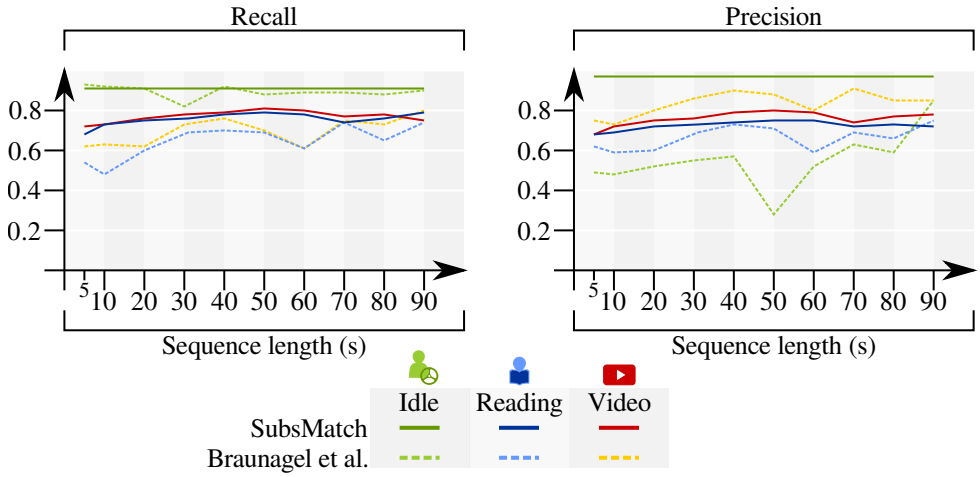


Fig. 8.1.4: The classification recall and precision of the prediction of the performed secondary task.

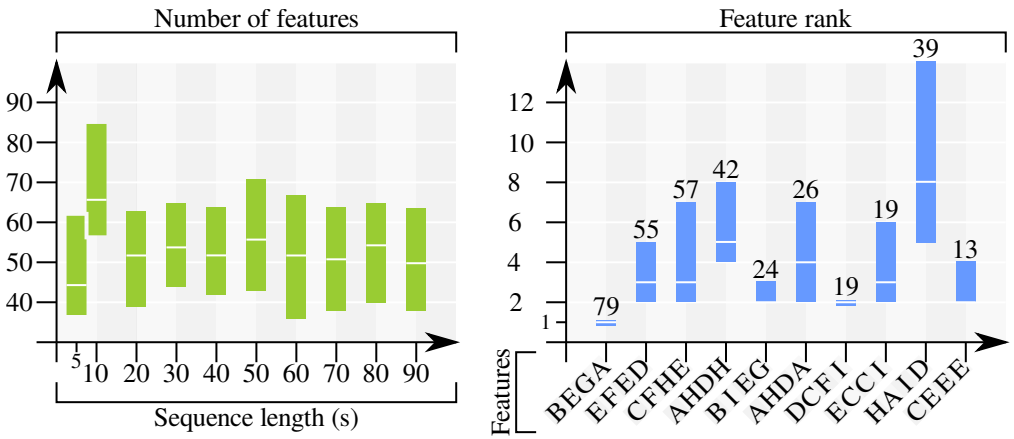


Fig. 8.1.5: The left plot shows the range of the number of selected features by the FCBF feature selection during the leave-one-out cross-validation on different analyzed sequence lengths. The white line indicates the median of the selected number of features. The right plot shows the ranking of a selection of features at a sequence length of 5 s, a word length of 4, and a symbol alphabet $\Sigma_{\text{SAX}} = \{A, \ldots, I\}$ ($|\Sigma_{\text{SAX}}| = 9$). The number on top of the bar indicates how often the word was selected as a feature during the leave-one-out cross-validation, e.g., the word `BEGA` was selected as a feature on 79 drivers of 82 total.

the second plot shows the selected features by FCBF for a sequence size of 5 s arranged by their ranking (the lower, the higher the ranking). The high ranked features reveal the significant patterns, i.e., the top-ranked word `BEGA`, which was selected for almost all 82 drivers. This particular feature correlates with the expected eye movement pattern while reading. The subsequence `BEG` describes a transition from left to right (word scanning), whereas the subsequence `GA` indicates a large saccade from right to left (newline). Moreover, 4 of the top 10 features shows patterns similar to reading, i.e., `BEGA`, `EFED`, `CFHE`, and, `DCFI`. However, features like `CEEE` describe a prolonged fixation at bin E and may indicate an idle phase, where the driver was just looking straight ahead of the road.

8.1.3 Limitations

The presented model was developed with the intention to perform online in an embedded car setup. However, considering the performed experiment, the eye-tracking system provided an eye signal at 25 Hz, meaning a runtime requirement of 40 ms to process a new eye-tracking sample. In this short period, the new word is extracted, filtered by the FCBF selected feature list, counted in terms of its frequency, and finally, the updated frequency vector is classified by the SVM. However, the size of the frequency vector crucial depends on the extracted word size, number of distinct symbols, and, finally, the selectivity of the FCBF.

Although the FCBF always efficiently reduces the number of characteristics to the desired level, it is advisable to enlarge the feature space if the number of possible words grows, e.g.,

due to a larger word size or a higher number of different symbols. A larger feature space enables the representation of a broader set of significant eye movement patterns. However, such larger patterns tend to occur less frequently, yet with greater variation. Hence, a restrictive FCBF feature selection may reject too many significant features and impairs the classification accuracy. Conversely, a large feature space may lead to increased requirements on memory and computing capacity; thus, the online criteria are not satisfied. This balance was achieved in the presented experiment with a word length of 4 and a symbol alphabet of size 9.

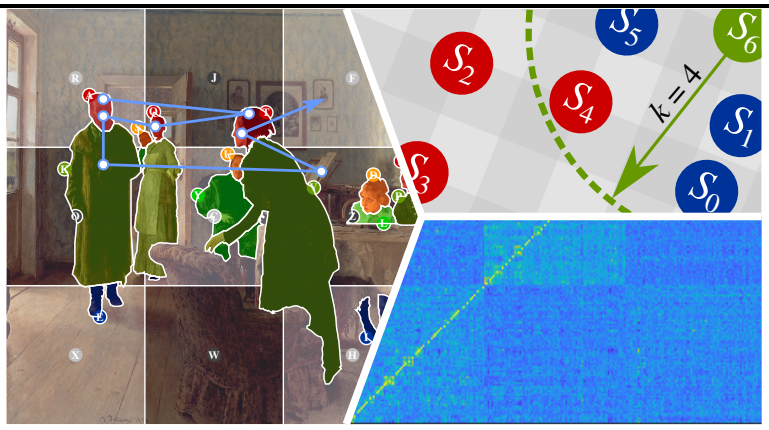
Furthermore, the classification of the distance vectors was deliberately omitted, rather the frequency vectors were classified directly, even though this would have promised better results. Yet, the calculation of the pairwise euclidean distance to a set of reference vectors is expensive in terms of memory consumption and processing time. However, the next Section 8.2 introduces a novel approach to process the scanpath similarity by extensive use of hashing: A MinHash based approximation of the Jaccard index [1].

8.1.4 Conclusion

This section introduced in companion with Section 7.2 a full processing stack to predict the driver's secondary task during a conditional driving scenario. The extensive evaluation base on a large-scale driving simulation study, performed in one of the most modern driving simulators in the world: the Mercedes Benz driving simulation platform [322]. The results were benchmarked against a state-of-the-art approach from Braunagel et al. [321]. The presented approach outperforms or rivals the state of the art approach, yet shows excellent online capabilities up to 25 Hz. A more in-depth glimpse into the feature selection provides insights into the significant eye movement patterns and helps to understand and tweak the model. Moreover, the analytic step provides reasoning for detected secondary tasks and makes the whole model more explainable.

The next Section 8.2 exploits the scanpath similarity task and provides a memory and processing time-efficient workflow for a Jaccard index approximation of two scanpaths.

8.2 A MinHash approach for fast scanpath classification



The previous Section 8.1 already demonstrated how powerful scanpath comparison is. Analyzing characteristic patterns reveal insights into the user's cognitive state and processing. Thus, scanpath analysis became part of the standard repertoire of eye-tracking researchers. However, as the length of the scanpath or the number of possible symbols increases, established methods for scanpath similarity become inefficient, both in terms of runtime and memory consumption. This section presents a MinHash approach for efficient scanpath similarity calculation. The approach shows competitive results in clustering and classification of scanpaths compared to established methods such as Needleman-Wunsch, yet at a fraction of the required runtime. Furthermore, with time complexity of $\mathcal{O}(n)$ and constant memory consumption, the presented approach is ideally suited for realtime operation or analyzing large amounts of data.

The following sections are supported by a prior publication from Geisler et al. at the conference on Eye-tracking Research & Applications (ETRA) in 2020 🏆 [1]. Section 8.2.1 elaborates on the methodology of a MinHash approach for scanpath classification. The subsequent evaluation in Section 8.2.2 evaluates the presented approach in terms of classification accuracy and runtime. Moreover, the evaluation benchmarks the MinHash based analysis against state-of-the-art string comparison methods, namely SubsMatch and Needleman Wunsch [197], [201], [208]. The final Sections 8.2.3 and 8.2.4 state the limitations and final remarks, respectively.

8.2.1 Method

MinHash is a widely used approach in the field of data mining and is mostly used for clustering and classifying documents [323]. This section elaborates on the adoption of the MinHash

🏆 Best paper award

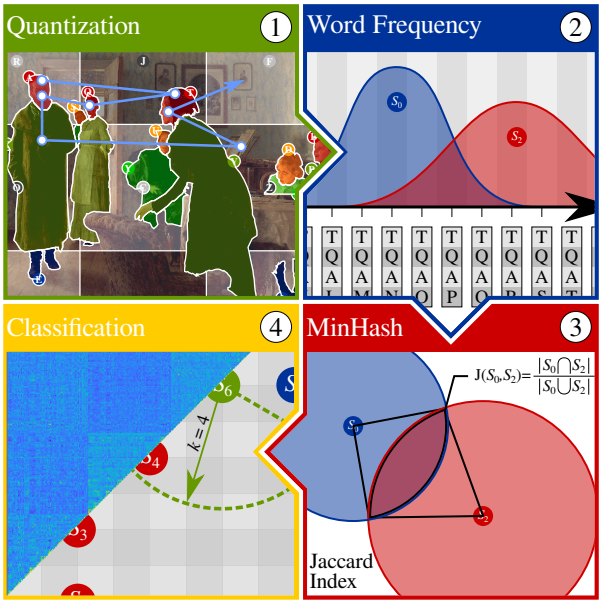


Fig. 8.2.1: Workflow of the proposed MinHash approach in 4 steps: (1) The recorded scanpath is quantified using ROIs on the stimulus. (2) Extraction and permutation of subsequences. (3) Approximate the Jaccard Index by randomly chosen pattern matches using MinHash (4) Extract features for scanpath classification.

approach to calculate scanpath similarities in order to cluster and classify them subsequently. Therefore, the scanpath is quantified to a string based on the stimulus' (semantic) ROIs, followed by a string subsequence extraction. The frequency of the subsequences is then used to estimate the Jaccard Index of two scanpaths using MinHash. The Jaccard Index of the scanpaths to each other is then used to classify new scanpaths.

The presented approach divides into four successive steps, as shown in Figure 8.2.1, which will be introduced and discussed in the following.

Step (1) Quantization First, the gaze signal is quantized to a string of symbols S. The quantification can be done using a static grid over the stimulus – or if available – based on ROIs. Every time a fixation falls into a grid cell or RoI, S gets expanded by the corresponding symbol. In addition to spatial representation, other attributes such as fixation duration, saccade length, or direction can also be used for quantification. For example, the fixation period is often represented as a repetition of the same symbol. Or, depending on the saccade length, an extra symbol is attached.

Figure 8.2.2 shows an exemplary quantification of a scanpath on the "Unexpected Visitor" image from the Yarbus experiment. It combines the grid for the background with semantic ROIs on for the foreground.

Step (2) Extract Word Frequencies The proposed method assesses the similarity of two scanpaths based on the frequency of visual scanpath subsequences. The following steps refer to a subsequence as word w. In the simplest case, a word of n consecutive symbols

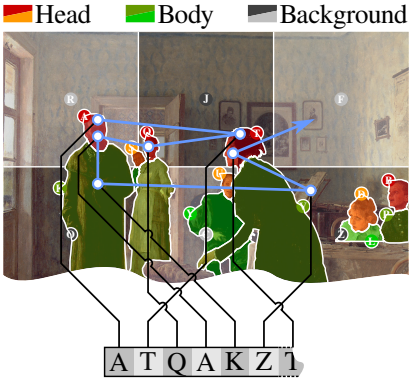


Fig. 8.2.2: Quantization of the scanpath shown in blue using semantic (red and green) and spatial ROIs (gray).

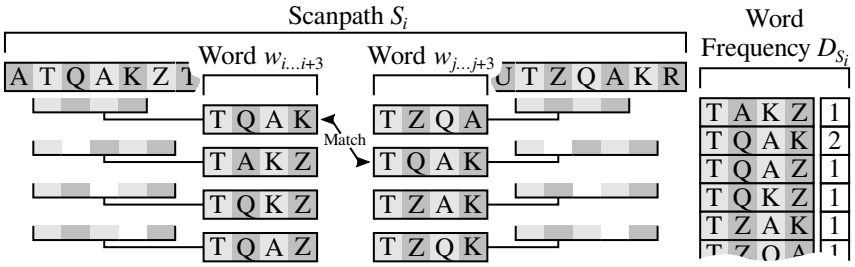


Fig. 8.2.3: Shows how the words with a length of four symbols and a maximum of one gap are extracted from the scanpath S_i through a sliding window. Each extracted word is counted in the Dictionary D_{S_i}. The sub sequence `TZQAK` is mapped to `TQAK` by the gap at `Z`, leading to a frequency of two for the word `TQAK` in D_{S_i}.

represents the sequence of fixation transitions. Yet, the symbol does not only represent the spatial region of the stimulus but it also includes semantic information, and the interpretation of such a word becomes more significant. For example, the illustrated in Figure 8.2.2, the word `ATQ` not only contains the information that the fixations have fallen into a certain area of the stimulus, but also implies that there were transitions between different head-related ROIs. These patterns are known to be particularly descriptive in terms of various classification tasks, such as predicting the currently performed task of the subject [67], [203], [209], [210].

To extract the words in a scanpath, the proposed method uses a sliding window approach. Moreover, it is also suggested to not only extract words of a fixed length. Hence, the proposed approach allows to extract words of different length and with gaps. Thus, scanpath subsequences are matched even with small variations. Figure 8.2.3 shows an example of the extraction of words with a length of four ($w_{\text{min}} = w_{\text{max}} = 4$) and one optional gap ($w_{\text{gap}}$). In addition, it is usually advantageous to scan the scanpath backward for words. Thus, subsequences can be depicted in the same word, even if they differ in one place or are reversed.

Each extracted word w of the scanpath S is counted in the Dictionary D_S in the form of a (chained) hash table with n_b buckets and the hash function $h(w) \longrightarrow [0, n_b)$. Until this step, the procedure is similar to SubsMatch. In the next step, SubsMatch would calculate the

distance of the dictionaries $D_{S_0}, \ldots, D_{S_n}$ with each other using any distance metric [197], or feed it directly as a feature vector in a machine learning process [208]. However, it is necessary to iterate over all words present in $D_{S_0}, ..., D_{S_n}$. Depending on the alphabet size n_a and extracted word lengths, this can become extremely costly: $\sum_{w_n=w_{\min}}^{w_{\max}} n_a^{w_n}$. This extreme case becomes more likely with increasing scanpath length. Instead, MinHash compares n_h random word frequencies. Therefore, the extracted words are additionally indexed in n_h hash tables $D_{H_0}, \ldots, D_{H_{n_h-1}}$ using the disjunct hash functions $h_0(w), \ldots, h_{n_h-1}$.

Algorithm 1 shows this step as a pseudo code. Extracting and counting the word frequency from S needs to iterate over the whole scanpath, which leads to a runtime $O(n)$. Accessing hash tables for a constant word length is a constant operation. The inner loop does not impact the runtime since n_h is constant.

Algorithm 1: Word frequency counting

```
Input: S
Data: D_{H_{0...n_h-1}}
Result: D_S
foreach w ∈ S do
    /* Increase the frequency of the word w in the scanpath
       dictionary D_S                                        */
    D_S(h_0(w)) ⟵ D_S(h_0(w)) + 1
    foreach D_{H_i} ∈ D_{H_{0...n_h-1}} do
        if h_i(w) ∉ D_{H_i} then
            /* Add word w to the global dictionary D_{H_i}  */
            D_{H_i}(h_i(w)) ⟵ w
```

Step ③ MinHash The underlying idea is that for an optimal (collision-free) hash function, h applies:

$$J_{D_{S_k},D_{S_l}} = P\left(h_{\min}\left(D_{S_k}\right) = h_{\min}\left(D_{S_l}\right)\right), \tag{8.7}$$

where $J_{D_{S_k},D_{S_l}}$ is the Jaccard index of the words in the dictionary D_{S_k} and D_{S_l}. $h_{\min}(D)$ extracts the minimal hash value of the words in the dictionary D. $P(\ldots)$ is the probability that the minimal hash value of the dictionary D_{S_k} is equal to the minimal hash value of D_{S_l}. In other words, the probability that the first word in the hash table of the dictionary D_{S_k} is equal to the first word in the hash table of the Dictionary D_{S_l} is equal to the Jaccard Index of D_{S_k} and D_{S_l}.

Then, the probability $P(\ldots)$ can be approximated by building many dictionaries $D_S^{(0)}$, $\ldots, D_S^{(n_h)}$ using different hash functions $h_0(w), \ldots, h_{n_h}(w)$:

$$P\left(h_{\min}\left(D_{S_k}\right) = h_{\min}\left(D_{S_l}\right)\right) \approx \sum_{i=0}^{n_h} \text{match}\left(D_{S_l}^{(i)}(0), D_{S_k}^{(i)}(0)\right) \cdot n_h^{-1}, \tag{8.8}$$

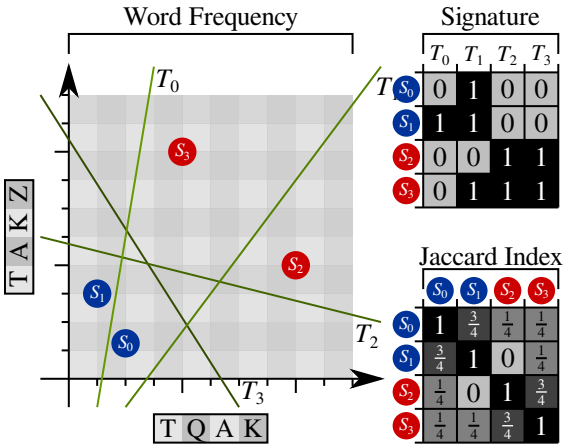


Fig. 8.2.4: Shows the application of random hyperplanes in the frequency domain of the words TAKZ and TQAK for four exemplary scanpaths $S_0, \ldots, S_3$. The Jaccard index is calculated from the Signature matrix, which indicates the side of the scanpath S_i regarding to a hyperplane T_i.

with

$$\text{match}\left(D_{S_l}^{(i)}(0), D_{S_k}^{(i)}(0)\right) = \begin{cases} 1 & \text{if } D_{S_l}^{(i)}(0) = D_{S_k}^{(i)}(0) \\ 0 & \text{otherwise} \end{cases}, \tag{8.9}$$

where $D^{(i)}(0)$ is the first entry of the hash table in the dictionary D using the hash function $h_i(w)$. However, in this case, words may occur multiple times in a scanpath. Therefore, the proposed approach deals with word frequencies. Instead of checking whether a word is present in both scanpaths, the match $(\ldots)$ function should estimate whether the frequency of both words matches and how it contributes to the Jaccard Index. For this purpose, the word frequencies are binarized by means of n_t randomly selected hyperplanes T. If both word frequencies are on the same side of the hyperplane, it is counted as a match, and the Jaccard Index is increased proportionally. Figure 8.2.4 visualizes this procedure using four hyperplanes $T_0, \ldots, T_3$. The closer the respective word frequencies are to each other, the more often they fall on the same side of the hyperplane, leading to a higher Jaccard Index. The match $(\ldots)$ function is, therefore, redefined as follows:

$$\text{match}\left(D_{S_l}^{(i)}(0), D_{S_k}^{(i)}(0)\right) = \sum_{j=0}^{n_t} \text{thresh}\left(D_{S_l}^{(i)}(0), D_{S_k}^{(i)}(0), T_j\right) \cdot n_t^{-1}, \tag{8.10}$$

where thresh $(\ldots)$ returns 1 if both word frequencies are on the same side of the hyperplane T_j, otherwise 0.

For the calculation of many scanpaths, it is necessary to build n_h dictionaries for each scanpath, which leads to extreme memory consumption. For the application on common hardware, it is advisable to do this calculation once for all scanpaths together: $D_{H_0}, \ldots, D_{H_{n_h-1}}$. These dictionaries also contain words that are not part of the scanpath pair D_{S_k}, D_{S_l}. Thus, it is necessary to iterate over D_{H_i} until the current word is contained in D_{S_k} or D_{S_l}; but the resulting overhead is small and reduces for longer scanpaths (see the runtime evaluation in Section 8.2.2).

Algorithm 2 shows the implementation of the proposed MinHash approach. In addition to the weighting by the hyperplanes, $w_{\text{weight}}^{|w|-w_{\min}}$ also includes the word length $|w|$. The idea is that longer words occur less often in two scanpaths, but are more meaningful about similarity.

c accumulates all weights and normalizes the estimated Jaccard Index at the end into the value range between 0 and 1. Please note that c also provides a good confidence measure about the approximation of the Jaccard Index $J_{D_{S_k},D_{S_l}}$.

Algorithm 2: MinHash

```
Input: D_{S_k}, D_{S_l}
Data: D_{H_{0...n_h-1}}
Result: J_{D_{S_k},D_{S_l}}
c ⟵ 0 and J_{D_{S_k},D_{S_l}} ⟵ 0
foreach D_{H_i} ∈ D_{H_{0...n_h-1}} do
    foreach w ∈ D_{H_i} do
        /* Only consider words that occur at least in one of the
           two scanpaths S_i and S_j.                              */
        if D_{S_k}(h_i(w)) > 0 ∨ D_{S_l}(h_i(w)) > 0 then
            /* Weighting of the match/mismatch found based on the
               word length.                                        */
            f ⟵ w_weight^{|w|-w_min}
            /* Count all weights for later normalization.         */
            c ⟵ c + f
            /* Apply random hyperplane thresholds T_0,...,T_{n_t-1} for
               local sensitivity.                                  */
            foreach T ∈ T_{0...n_t-1} do
                if D_{S_k}(h_i(w)) > T = D_{S_l}(h_j(w)) > T then
                    /* S_k and S_l on the same side of the hyperplane
                       T. Increase Jaccard Index approximation
                       J_{D_{S_k},D_{S_l}}.                        */
                    J_{D_{S_k},D_{S_l}} ⟵ J_{D_{S_k},D_{S_l}} + f · n_t^{-1}
            break
J_{D_{S_k},D_{S_l}} ⟵ J_{D_{S_k},D_{S_l}} · c^{-1}
```

Step ④ Classification The classification of an unknown scanpath S_{test} can be realized over the Jaccard Index to the training scanpaths $\mathbf{S}_{\text{train}} = (S_0, ..., S_n)$ and their related class labels $\mathbf{C}_{\text{train}} = (C_0, ..., C_n)$. Then the majority class label of the k most similar training scanpath is predicted as C_{test}: a kNN classifier.

However, considering the similarity matrices in the appendix Figure A.4 (ETRA challenge 2019), it is apparent that one class of scanpaths (Natural) does not form a distinct cluster. Rather, the similarity of the scanpaths seems to be equally similar to all others. The prediction of scan paths of this class by the k nearest neighbors would likely fail. As long as there is only one such weakly pronounced class, however, this can be handled with a small trick. Instead of choosing the k most similar scanpaths, the k scanpaths with the

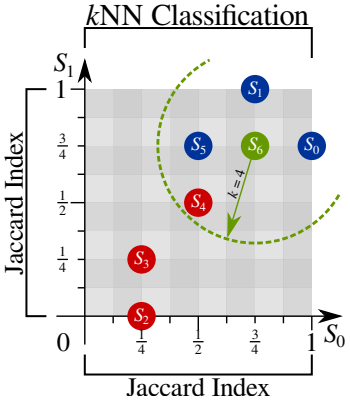


Fig. 8.2.5: kNN classification using the Jaccard Index of the scanpath S_0 (horizontal axis) and S_1 (vertical axis) to the scanpaths $S_0, \ldots, S_6$. The new scanpath S_6 would be classified as blue, since three of the four nearest neighbors correspond to this class.

most similar Jaccard Indices are chosen. Therefore, a similarity matrix $\mathbf{J}_{\text{train}}$ overall training scanpaths is calculated. Each row and column of the matrix $\mathbf{J}_{\text{train}}$ corresponds to one training scanpath. Thus, the element at the row r and column c represents the Jaccard Index $J_{D_{S_r}, D_{S_c}}$. Next, the Jaccard Index of the scanpaths S_{test} is calculated to all training scanpaths $\mathbf{S}_{\text{train}}$: $\mathbf{J}_{\text{test}} = J_{D_{S_0}, D_{\text{test}}}, \ldots, J_{D_{S_n}, D_{\text{test}}}$. For the classification step, the k most similar (euclidean distance) rows in $\mathbf{J}_{\text{train}}$ to $\mathbf{J}_{\text{test}}$ are determined. Since each row in $\mathbf{J}_{\text{test}}$ corresponds to a training scanpath, and thus to Class label, the predicted class label can be determined by a majority decision. Figure 8.2.5 shows the procedure with regard to the Jaccard similarity of the seven scanpaths $S_0, \ldots, S_6$ to S_0 and S_1.

Yet, this approach can also be used with other classifiers instead of kNN. Algorithm 3 shows an abstract workflow to train and classify scanpaths based on the Jaccard Similarity. The functions fit () and predict () can be exchanged with any classification method that supports multidimensional features, which is beneficial, especially if the classification must be in realtime. kNN has the disadvantage of a lazy-learning approach. Meaning, the modeling does not take place during training, but at the time of prediction. An eager-learning approach, like the SVM, creates a model (hyperplane) while training and does not run overall training data for the prediction. The evaluation showed favorable outcomes using a linear SVM and tree classifier. Yet, the performance of these methods depends strongly on its parametrization. Hence this section is limited to the single parameter approach kNN in the following to keep the evaluation neat and clear.

8.2.1.1 Implementation details

In the following gives some short remarks to the implementation details, in order to optimize the traceability in a practical application of the presented approach:

1. MinHash makes extensive use of hash tables. In most standard libraries, they are implemented as chained [324], which means that n_b buckets are reserved. The hash function assigns any word to one of these buckets. Collisions are solved by maintaining a linked list in each bucket. Additionally, a second linked list is maintained, referencing active (non-empty) buckets. This enables fast iteration over the contained elements. However, it implicates two issues: (i) no permutation inside a bucket $\rightarrow$ the permutation is limited by n_b. It is therefore advisable to select n_b large enough

Algorithm 3: Classification

Input: $S_0, \ldots, S_n$ and $C_0, \ldots, C_n$ $S_{\texttt{test}}$

Result: $C_{\texttt{test}}$

/* The feature vector to train the classifier is the square matrix J of the pairwise Jaccard Index $J_{D_{S_k}, D_{S_l}}$ of the training scanpaths $S_0, \ldots, S_n$, respective their dictionaries $D_{S_0}, \ldots, D_{S_n}$. Please note $J_{D_{S_k}, D_{S_l}} = J_{D_{S_l}, D_{S_k}}$. */

$$\texttt{J}_{\texttt{train}} \longleftarrow \begin{pmatrix} 1 & J_{D_{S_0}, D_{S_1}} & \cdots & J_{D_{S_0}, D_{S_n}} \\ J_{D_{S_1}, D_{S_0}} & 1 & \ddots & \vdots \\ \vdots & \ddots & 1 & J_{D_{S_{n-1}}, D_{S_n}} \\ J_{D_{S_n}, D_{S_0}} & \cdots & J_{D_{S_n}, D_{S_{n-1}}} & 1 \end{pmatrix}$$

$\texttt{C}_{\texttt{train}} \longleftarrow \{C_0, \ldots, C_n\}$

$M \longleftarrow \texttt{fit}\,(\texttt{J}_{\texttt{train}}, \texttt{C}_{\texttt{train}})$

/* Analogous to the training input $\texttt{J}_{\texttt{train}}$, the test vector is assembled as the pairwise Jaccard Index $\texttt{J}_{\texttt{test}}$ of the training scanpaths $\texttt{S}_{\texttt{train}}$ to the test scanpath $S_{\texttt{test}}$ */

$\texttt{J}_{\texttt{test}} \longleftarrow \begin{pmatrix} J_{D_{S_0}, D_{S_{\texttt{test}}}} & \cdots & J_{D_{S_n}, D_{S_{\texttt{test}}}} \end{pmatrix}$

$C_{\texttt{test}} \longleftarrow \texttt{predict}\,(M, \texttt{J}_{\texttt{test}})$

depending on the number of different words in a scanpath. (ii) The order of the bucket activation is reconstructed by the active bucket list. Together with (i), the insertion order is partly reconstructed when iterating. For algorithm 2, it must be assured to iterate over the buckets in their logical order, not in their activation order.

2. The random hyperplanes in algorithm 2 can be efficiently approximated by cosine similarity [325].

3. The proposed approach benefits from the fact that almost all steps can be parallelized: The extraction of words by using different threads for each window size and gap position. The MinHash by using dedicated threads for each permuted dictionary D_{H_i}. And the pairwise calculation of the Jaccard Index.

8.2.2 Evaluation

The evaluation does not focus on the absolute classification accuracy, but rather on how close the classification accuracy of MinHash is to the state-of-the-art, at a shorter runtime. The evaluation comprises a benchmark of MinHash with SubsMatch and Needleman-Wunsch. All three approaches are string-based methods and thus use the same input features (quantized scanpath from step ①). In addition, all three approaches calculate similarities or distances between strings, which makes it possible to use a uniform classification method (kNN from step ④). This ensures that the actual novelty (step ③) can be evaluated and ranked as isolated as possible to the existing approaches.

The compared approaches were evaluated on two different datasets: The ETRA challenge dataset from 2019, and a self-recorded Yarbus dataset with three different tasks. Furthermore, the runtime behavior for different scanpaths lengths and the number of subjects was investigated using synthetic data.

8.2.2.1 ETRA challenge 2019

The dataset from the ETRA challenge 2019 [326], [327] consists of 345 task-related recordings. 115 each for free viewing a natural scene (15 different stimuli), image puzzles (18 different stimuli), and find waldo (15 different stimuli). The gaze signal was quantized by a fixed 7×7 grid over the stimulus. Then, all sequences of consecutive repeating symbols shorter than 8 where deleted (8 consecutive symbols correspond to $\sim$100 ms @ 75 Hz), since they were considered too short as a valid fixation. Sequences longer than or equal to 8 symbols were reduced by the factor of 8, but truncated to maximal 3 consecutive symbols. E.g.:

$$\underbrace{\texttt{EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE}}_{32\times\texttt{E}\rightarrow\texttt{EEE}}\texttt{AA}\underbrace{\texttt{TTTTTTTTT}}_{9\times\texttt{T}\rightarrow\texttt{T}}\texttt{ETE}\underbrace{\texttt{RRRRRRRR}}_{8\times\texttt{R}\rightarrow\texttt{R}}\texttt{EE}\underbrace{\texttt{AAAAAAAAAAAAAAAAA}}_{17\times\texttt{A}\rightarrow\texttt{AA}}\texttt{TT}$$

Is reduced to: `EEETRAA`. This leads to an enormous reduction of the scanpaths. However, the average scanpath length is still 313 symbols (min. 173, max. 500).

MinHash has been configured with $w_{\text{min}} = 2$, $w_{\text{max}} = 3$, a valid gap $w_{\text{gap}} = 1$, and a base weight of $w_{\text{weight}} = 1.2$. The evaluation was conducted as 345-fold cross-validation using Matlab's `crossvalind` function to partition the training and test set. The classification was performed as described in Section 8.2.1 step ④ using kNN ($k = 4$) in the Jaccard Index domain, using Matlab's `fitcknn` function. The prediction target was the performed task of the subject (natural viewing, puzzle, where's Waldo) across all stimuli.

The appendix Figure A.4 shows the results for different numbers of hash functions $n_h = \{64, 128, 256, 512\}$, and for SubsMatch (Needleman-Wunsch was runtime conditionally not applicable). As the number of different hash functions increases, the accuracy of the classification increases until it saturates at about 88 % at $n_h = 256$. SubsMatch achieves an accuracy of 91 %. However, MinHash takes only half of the processing time for $n_h = 256$ by almost the same performance as subsmatch.

8.2.2.2 Yarbus

This dataset is much more challenging in terms of classification performance compared to the ETRA 2019 challenge dataset. The recording includes 23 subjects. 10 images from the Defending Yarbus [328] were presented to each subject in randomly permuted order for 10 seconds. For each image, the subject was asked to solve one of three different tasks: 1) estimate ages of the people, 2) remember clothes, and 3) estimate how long the visitor had been away.

The scanpaths were created using semantic ROIs with a background grid of 3×3. Therefore heads, bodies, and other prominent image areas were marked with rectangles. In order to compensate for noise and inaccuracies in the gaze signal, and to eliminate overlapping ROIs, the ROIs were grown until they encountered each other or reached a maximum size (see

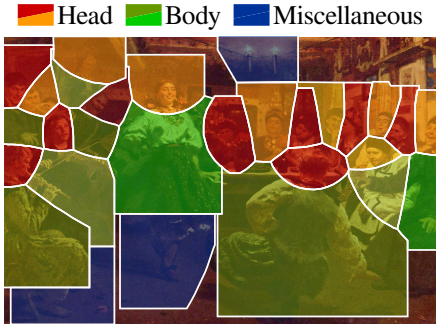


Fig. 8.2.6: Grown semantic ROI of heads, bodies, and various scene parts on one of the Defending Yarbus [328] stimuli.

Figure 8.2.6). The resulting scanpath was reduced similar to the ETRA 2019 challenge dataset, but with a reduction factor of 6 (60 Hz eye-tracker). This resulted in 10×23 scanpaths with an average length of 26 symbols (min. 5, max. 36).

Analogous to the ETRA 2019 Challenge, a cross-validation with $w_{\text{min}} = 2$, $w_{\text{max}} = 3$, a valid gap $w_{\text{gap}} = 1$, a base weight of $w_{\text{weight}} = 1.2$, and kNN with $k = 4$ was performed. The appendix Figure A.5 shows the classification performance of MinHash with $n_h = 128$, SubsMatch, and Needleman-Wunsch. The prediction target was the task performed per stimulus. It is noticeable that this challenge is much more difficult than the ETRA 2019 challenge. The best result was achieved by SubsMatch with an accuracy of 59 % and a chance of 33 %. However, this evaluation clearly shows the limits of the MinHash approach. For such short scanpaths and few subjects, MinHash is not able to play out its speed boost significantly: Even Needleman-Wunsch is faster. At the same time, due to its design, MinHash can never achieve better accuracy than SubsMatch.

8.2.2.3 Runtime

MinHash has a complex runtime behavior. In theory, MinHash's runtime is $O(n)$, since it needs n operations to scan the input scanpaths and to build up the hash tables. However, since all considered algorithms have to read the input with a runtime of $O(n)$, the runtime evaluation only examine the part of the actual similarity calculation. In this case, the runtime of MinHash is constant for large n.

The runtime evaluation was performed on a desktop computer with moderate hardware: i5-4590 @ 3.30 GHz CPU and 16 GB memory. All methods were tested using a single thread. The evaluation was done by randomly generated scanpaths since it reflects the worst-case scenario for the MinHash and SubsMatch algorithm. Therefore, a uniform distributed random sequence of symbols of a given alphabet, and a fixed scanpath length was generated. Thus, each word has the same likelihood to be part of the scanpath, which leads to huge hash tables. For short but many scanpaths, MinHash suffers from many different words in the dictionary, but a low probability of finding them in a single scanpath. Therefore, it iterates through the Dictionary until the word appears in one of the compared scanpaths (see algorithm 2 line 4). This event can be prevented by creating n_h dedicated hash tables for each scanpath pair. These hash tables contain only words that occur in at least one of the two scanpaths. Instead of iterating through D_{H_i}, the first element can be accessed directly. However, this approach is not technically feasible in most cases, in terms of

memory consumption. In addition, the overhead would remain linear during indexing but still increases significantly.

Additionally, it can be shown that the actual number of iterations is comparatively low for large scanpaths. The probability that q iterations on the global dictionary D_{H_i} are necessary in order to obtain a word that is part of the scanpath dictionaries D_{S_k} or D_{S_l} can be described as a geometric distribution with an expected value of [329]:

$$E\left[P\left(X=q\right)\right] = (1-p)\cdot p^{-1}, \tag{8.11}$$

where p is the probability of finding a word that is part of one of both scanpaths at a random position in D_{H_i}:

$$p = \left|D_{S_k} \cup D_{S_l}\right| \cdot \left|D_{H_i}\right|^{-1}, \quad \text{whereby} \quad D_{H_i} = \bigcup_{j=0}^{n} D_{S_j}, \tag{8.12}$$

which resolves to:

$$E\left[P\left(X=q\right)\right] = \left|D_{H_i}\right| \cdot \left|D_{S_k} \cup D_{S_l}\right|^{-1} - 1. \tag{8.13}$$

Further, for long scanpaths applies:

$$\lim_{|S_k,S_l|\to\infty} \left|D_{S_k} \cup D_{S_l}\right| = \left|D_{H_i}\right|, \tag{8.14}$$

respectively:

$$\lim_{\left|D_{S_k}\cup D_{S_l}\right|\to\infty} E\left[P\left(X=q\right)\right] = 0, \tag{8.15}$$

under the assumption that a scanpath is a random process of symbols. In fact: p is the Jaccard Index of the union of the Dictionaries D_{S_k} and D_{S_l} to the global dictionary D_{H_i}.

Figure 8.2.7 shows the runtime of the Needleman-Wunsch, SubsMatch, and MinHash algorithm for different scanpath lengths. For two scanpaths, the duration of MinHash is almost constant for different n_h but increasing scanpath length $|S|$. However, the runtime behavior at 64 scanpaths shows an interesting effect. First, the runtime increases for larger scanpaths. This is expected,since the number of words in the dictionary grows faster than the likelihood of a word to be in the scanpath. However, after a certain point, the runtime decreases and saturates on a certain level. This is a remarkable behavior for an algorithm, and is attributed to two competing effects: First, the dictionary size saturates at the level where all possible words are indexed. At this point, the negative impact of larger scanpaths also saturates. Second, the likelihood of a word to be part of a scanpath increases, and thus, fewer iterations for each hash permutations are necessary to find a word that is part of at least one of the compared scanpaths. Figure 8.2.8 shows this effect more emphasized. Comparing more than 32 scanpaths, MinHash is slower for short scanpaths (256 symbols) than for long scanpaths (512 symbols).

The first effect also applies to SubsMatch. Yet, SubsMatch does not use a global dictionary but uses the dictionaries of the respective scanpaths: $D_{H_0} = D_{S_k} \cup D_{S_l}$. With increasing

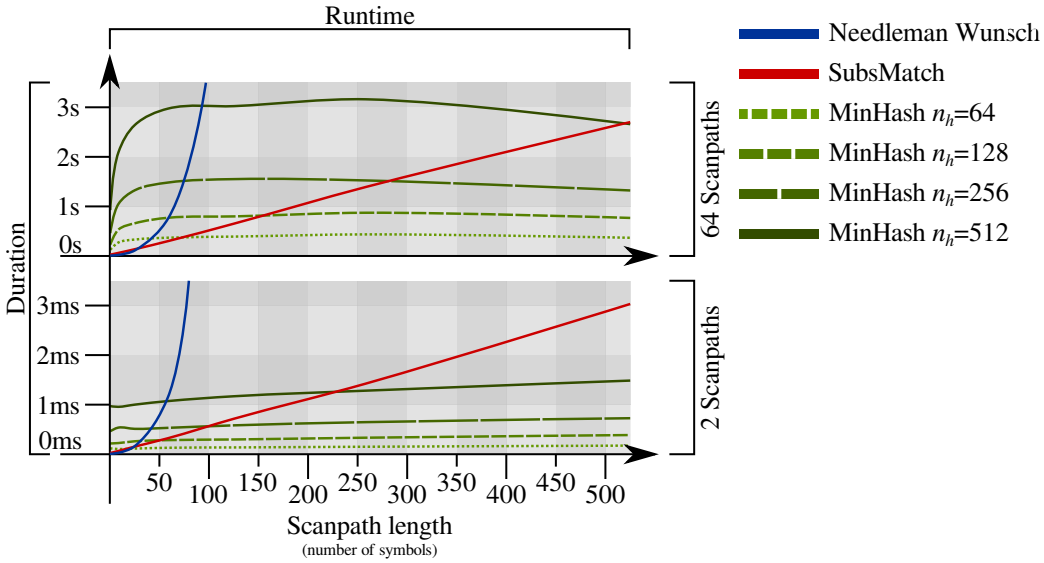


Fig. 8.2.7: Shows the runtime behavior of the MinHash approach as well as of SubsMatch and Needleman-Wunsch with increasing scanpath length. Needleman-Wunsch is $O\left(n^3\right)$ (or $O\left(n^2\right)$ as Levenstein distance). Assuming that a scanpath is a random process, the runtime of SubsMatch and MinHash saturates for long scanpaths.

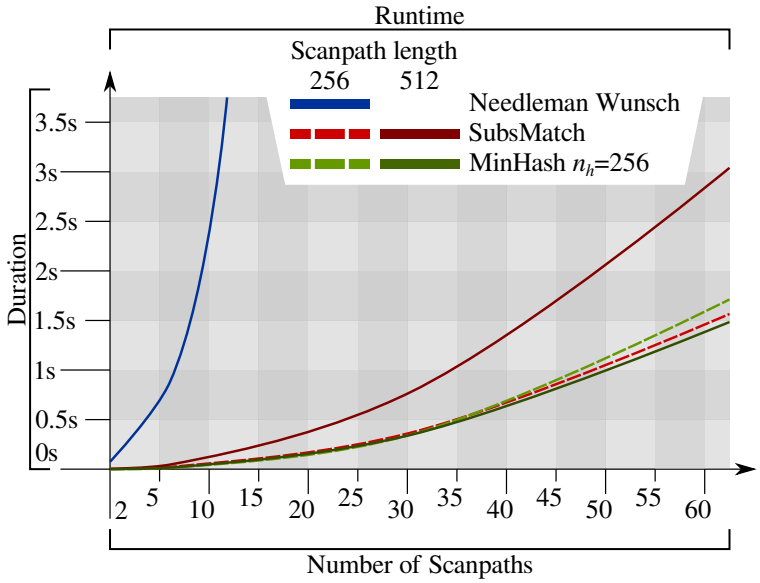


Fig. 8.2.8: Shows the influence of the increasing number of scan paths for a fixed scanpath length.

scanpath size, the probability increases that D_{H_0} contains all words, and the runtime saturates.

8.2.3 Limitations

The evaluation of the Yarbus dataset shows that MinHash is not suited for every dataset. Only if the scanpath length in relation to the number of scanpaths is large enough, MinHash becomes really fast compared to SubsMatch (and to Needleman-Wunsch anyway). It should also be noted again that MinHash approximates the Jaccard Index by testing n_h words, and therefore cannot achieve the classification performance of SubsMatch. SubsMatch calculates the Jaccard Index completely over all extracted words.

In addition, despite the global dictionary, the memory requirements for many words can become very extensive. For the calculation of the similarity matrix $\mathbf{J}_{\text{test}}$ in the ETRA 2019 challenge evaluation, $\sim$170 MB of the working memory is required with $n_h = 256$, $w_{\text{min}} = 2$, and $w_{\text{max}} = 3$. For the calculation with $w_{\text{max}} = 4$, there are already more than 500 MB required.

8.2.4 Conclusion

This section presented and evaluated a method based on MinHash that efficiently estimates the Jaccard Index in a quantized scanpath with large length and word variety. On the Etra 2019 challenge dataset, the evaluation demonstrated that MinHash almost achieves the classification performance of SubsMatch, yet at a significantly lower runtime. Moreover, the evaluation also showed that this runtime advantage increases with increasing input size. Thus, MinHash is particularly suitable for data mining in large quantities of long scanpaths. This provides the community a tool that allows encoding extensive information when quantizing scanpaths without the exploding alphabet and scanpath size affecting the classification runtime. At the same time, the Jaccard Index approximation of MinHash is mathematically reasoned and cleanly proven, which supports the validity of derived propositions.

The C++ implementation of MinHash, SubsMatch, and Needleman-Wunsch is freely available (including Matlab interface) on a public GIT repository. Linux users may also use the ready to run binaries: ↗ `https://atreus.informatik.uni-tuebingen.de/geislerd/pe-tools-scanpath`

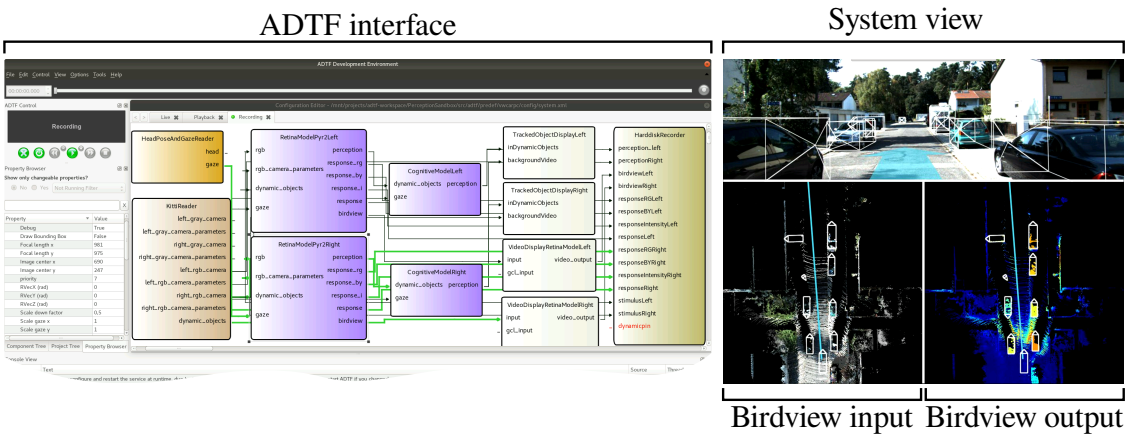


Fig. 8.3.1: Integration of the Perception sandbox into ElectroBit's ADTF. The left figure part shows the modeled processing graph using mixed components from the Perception sandbox and ADTF. The depicted graph processes the KITTI dataset along with a gaze signal to predict the user visual perception [109]. The core is formed by the retinal perception model presented in Section 6.2 [4]. The right part of the figure shows the generated (debug) output. The top view shows the fused system view of tracked objects and the driver's gaze , projected into one of the cars scene cameras. The two bottom view show the system view in a bird perspective including the tacked traffic objects (white bounding boxes), the drivers gaze (light blue line), as well as the 3D RGB voxels and the predicted perceivability on the left and right view, respectively.

8.3 Perception sandbox

The Perception sandbox was started as a successor the Saliency sandbox presented in Section 5.1. However, the Perception sandbox aggregates the approaches presented in Section 5.1, 6.1, and 6.2. Moreover, developed for as real-time signal processing framework, it provides interfaces to Matlab's Simulink framework and ElectroBit's Automotive Data and Time-Triggered Framework (ADTF). Thus, the Perception sandbox allows to run all the presented algorithms in a prototype car for advanced evaluation in driving and compound driving system. Furthermore, its well-conceived design allows pre-, online-, and post-processing of experiments to cover the full pipeline of developing novel – gaze-driven – models and applications.

Like the Saliency sandbox, the Perception sandbox implements a graph processing paradigm. Nodes represent dedicated processing units, e.g., algorithms, reader and writer, or user interfaces. Edges connect the output of a node to an input of one or multiple other nodes. The graph processor requests the node's output regarding a given timestamp, whereby the node's inputs are traced and processed to the given timestamp. Thus, the Perception sanbox implements a time triggered paradigm, similar to Matlab's Simlunk, yet in contrast to ElectroBit's ADTF (data triggered). The graph structure enables maximal parallelism

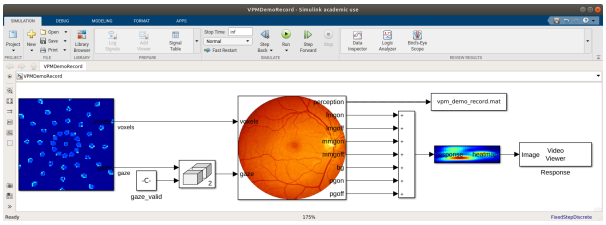


Fig. 8.3.2: Simulink model including the Perception sandbox's implementation of the biophysiological driven retina model presented in Section 6.1 [3]. Similar to the ADTF interface, the Perception sanbox's simulink interface allows to mix components with third-party modules. Morover, Simulink modules may used directly Matlab.

and result reuse, combined with flexible design and high-level application design possibility. Moreover, the Perception sandbox is straight forward extendable to additional nodes by implementing the abstract node interface.

Figure 8.3.1 shows an examplary processing pipeline, modeled in ADTF, yet processed by the Perception sandbox backend to predict a drivers perception likelihood of traffic objects based on the functional retinal perception model presented in Section 6.2. Similar pipeline, yet processing the evaluation of the biophysiological model presented in Section 6.1 and using Matlab's simulink interface is provided in 8.3.2. A substantial part of the framework is publicly open and is available through the University of Tübingen's GIT repository: `https://atreus.informatik.uni-tuebingen.de/geislerd/visualperceptionmodel`

The Perception sandbox was designed particularly for the use in vehicle prototypes and paved an essential milestone towards the joint patent of the thesis author and the long-term industrial partner Volkswagen AG [20]. The perception sandbox closed up the thesis by a practical application of the vast of the presented models and algorithms. The next section provides the thesis' final statements and conclusion.

CHAPTER 9

Conclusion and outlook

The last decade was characterized by progressively more advanced methods for video-based eye-tracking. The steady improvement and broader availability of sensors, coupled with increasingly advanced algorithms, have enabled the tracking of the user's line of sight at high accuracy and robustness. At the same time, video-based eye-tracking is minimally invasive compared to other eye-tracking technologies, e.g., EOG or scleral coil tracking. Thus, video-based eye-tracking emerged as the go-to method for advanced gaze-driven applications. In driver-monitoring, e.g., a remote eye-tracking system can be seamlessly integrated into the car's cockpit to provide a fully transparent interface for human-car interaction. This increases driver acceptance as the sensor is not directly attached to the user – in contrast to head-mounted eye-tracking systems – and simultaneously assures constant and controllable image acquisition conditions. Furthermore, an on-board eye-tracking sensor enables the straightforward fusion of the user's line of sight with the vehicle's environmental sensors, e.g., radar, cameras, or lidar. Advanced driving assistance may assess the driver's line of sight to, e.g., warn for unperceived hazardous situations, attention-aware interaction between driver and car system, or to ensure the driver is in the loop before handing over the driving task in highly automated driving.

However, head-mounted eye-tracking provides the potential of higher accuracy and may pave the way to ubiquitous eye-tracking. Lightweight hardware, directly mounted on the user's head, enables eye-tracking beyond a particular application domain. Such systems may contribute to advance digital personal assistance systems over the whole day. Moreover, coupled with AR technology, e.g., Microsoft's Hololens, head-mounted eye-tracking may enable responsively gaze-aware HUD designs [4], [16].

Yet, all these scenarios require accurate and robust detection of the user's line of sight. This thesis provided an in-depth overview of cutting edge algorithms for head-mounted and remote eye-tracking in Section 2.1 and 2.3. Moreover, three novel approaches were presented, pushing the state-of-the-art towards an even more accurate and robust extraction of the user's line of sight, even in particularly challenging conditions, such as varying lightning or excessive head pose offsets. Section 3.1 introduced a pupil detection approach based on CBF and random ferns for head-mounted eye-tracking [10], Section 3.2 provided a cycle GAN framework for eye image segmentation, refinement, and generation [9], and

Section 4.1 introduced a robust glint-based eye detection for remote eye-tracking designed for particularly challenging conditions [5]. Moreover, Section 4.2 provided an extensive review and evaluation of state-of-the-art pupil detection algorithms designed for remote eye-tracking [13].

The tracked line of sight provides the mere focus of the user's visual attention in the scene. Yet, the user's visual perception extends far beyond a straight line in the scene. The human retina perceives the scene stimulus across a field of view from 60° nasal to 107° temporal and 70° superior to 80° inferior. Over this extent, approximately 6 million cones (predominantly in the foveal area) and 120 million rods (predominantly in the periphery) react to the incident light and modulate the stimulus as biochemical, electrical potential. The subsequent neural network extracts and emphasizes salient visual features and feedforwards them to the optical nerve. Thus, to exploit the full potential of eye-tracking in HCI and HRI, it is necessary to develop the mere tracking of the user's line of sight to a comprehensive understanding of the user's scene perception. The models presented in Part II aim to replicate the retinal stimulus processing in order to predict the scene's visual perceivability from a user perspective. Thus, Section 5.1 introduced an extensive framework to model and deploy state-of-the-art saliency maps, providing a unified interface to implement custom processing steps and recombination of well-established saliency map function units. Moreover, Section 5.2 introduced a gaze-driven saliency approach by incorporating the user's line of sight into the processing of the GBVS.

Furthermore, Section 6.1 and 6.2 present two sophisticated models for biophysiological accurate and gaze-driven prediction of the user's retinal processing of the visual stimulus. Therefore, the retinal signal modulation is replicated and parametrized w.r.t. the scene's eccentricity towards a tracked line of sight – the user's egocentric foveal center. Thus, these models incorporate the stimulus' extent into the peripheral field of view, while also modeling distinct spatial and temporal retinal perception capabilities according to the affected retinal region. The retinal model introduced in Section 6.1 provides unprecedented accuracy and detailed reproduction of bio-physiological retinal processes. The evaluation showed that in specific application domains, the model even enables the prediction of the next fixation target by assessing the peripheral stimulus [3]. This is a reliable indicator of the quality of the model's prediction of the peripheral, which might advance other research areas, e.g., foveated rendering. In this discipline, only the foveal area is rendered with high resolution to save bandwidth and processing capacities [330]. However, a major challenge is to ensure that the new foveal area is rapidly identified and rendered with high resolution after a saccade. A reliable prediction of the next most likely fixation target by the proposed retina model may provide the vital time advantage for a foveal rendering that is indiscernible to the user.

In the context of advanced driving assistant systems and driver monitoring, understanding peripheral vision is crucial to predict the driver's situation awareness. For instance, experienced drivers fixate the vanishing point of their trajectory most of the time, yet track traffic relevant objects in their peripheral field of view [105]. Thus, a mere evaluation of the driver's line of sight likely underestimates the driver's actual situational awareness of relevant traffic objects. Hence, the retinal processing model presented in Section 6.2 evaluates the user's field of view as the bio-physiological driven approach in Section 6.1, while being optimized in terms of runtime and resource consumptions for an online operation mode in a car-setup.

The evaluation reveals that the model successfully predicts the perceivability of task-relevant changes on the visual stimulus [4].

However, retinal perception is the first step of the user's visual processing pipeline. A subsequent highly developed cognitive apparatus filters relevant visual features passed through the optical nerve. Then, the features are successively combined and abstracted to more expressive and valuable information, e.g., objects, shapes, or trajectories, before they are presented to the conscious perception. The stimulus' semantic significance drives this filtering and abstraction step w.r.t. top-down features such as the currently performed task, intention, experience, cognitive load, fatigue, and other factors. Naturally, the manifestation of such factors in the actual conscious perception is intrinsic to each individual and likely too complex to be generalized.

Part III elaborated on eye features that enable the prediction of the aforementioned factors. In particular applications, e.g., driving, these features enable a prediction of current task-related cognitive conditions and enable a more detailed prediction of the user's visual situation awareness. Thus, Section 7.1 and 7.2 presented two novel approaches for cognitive load and task prediction by pupil dilation and fixation-saccade statistics, respectively [7], [12]. Both models were developed for an online operation mode in a driving setup. The subsequent Sections 8.1 and 8.2 elaborated deeper into task prediction by two novel scanpath based approaches [1], [11]. The automatic detection of the driver's performed (secondary) task is a particularly valuable feature to weight and filter visual features extracted by a retina model in Part II. Thus, a driver performing a high load secondary task, e.g., writing mail, likely shifted their visual attention bias apart from driving-related visual features to writing-related visual features. In such a scenario, an advanced driving assistant system may assess even salient – retinal perceivable – traffic objects as likely unperceived by the driver. Similarly, if the system detects a focused and experienced driver, it becomes more likely that a traffic object was perceived, even if it was not particularly salient in the user's field of view. An advanced driver assistance system may incorporate both modalities – provided by the retinal and the cognitive user model – to assess whether the driver requires assistance in particular situations. For example, a conditionally automated driving system may evaluate if the driver is capable of taking over the vehicle control safely.

The presented models in this thesis allow a comprehensive and in-depth assessment of the user's visual perception, driven by eye-tracking and visual scene evaluation. All presented models were developed for online application mode, in particular in the automotive domain. Moreover, Section 8.3 presented an extensive framework that encapsulates most of the presented models into a straightforward usable signal processing framework, e.g., Electrobit's ADTF, or Matlabs's Simulink. Moreover, this framework was used to implement and test the presented algorithms into a prototype car platform and provided a major step into the join patent development of the thesis author and the industry partner Volkswagen AG [20].

The European New Car Assessment Programme (Euro NCAP) Roadmap 2025 lists driver monitoring as an urgent safety feature required for a five-star ranking from 2020 [331]. Thus, video-based driver monitoring is about to become broadly available on the mass market for modern cars. Hence, Daimler recently announced that the new 2021 Mercedes S-Klasse would be equipped with a driver monitoring camera in the instrument cluster to provide eye-tracking technology [332]. At the same time, companies such as Smart-Eye and

Tobii, as well as traditional automotive suppliers such as Bosch and Continental, are ready to launch video-based driver monitoring modules with eye-tracking capabilities for the mass market [333]–[336]. This very dynamic development of user interfaces in the automotive context makes this thesis particularly relevant for the ongoing and future development of next-generation driving assistance systems.

However, one of the biggest challenges of the presented perception and cognitive models is their evaluation and parameterization for different applications and scenarios. In particular, driving is a versatile and complex activity. Drivers may provide different levels of experience, expertise, and techniques, whereas extremely diverse traffic scenarios provide enormous variability in real-world conditions. Thus, the next step towards video- and eye-tracking-based driver monitoring systems in next-generation driving assistance systems requires precisely specified application domains and automatic detection of the system limits. While the bottom-up retinal models presented in Part II generalize well as they replicate bio-physiological findings, top-down interpretation, e.g., through a subsequent NN as presented in Section 6.1 and 6.2 requires massive training and evaluation data [3], [4]. However, the imminent availability of eye-tracking technology in mass-produced cars may provide the requisite data basis.

Cognitive eye features, as presented in Part III, i.e., pupillometry, eyelid closure, and fixation and saccade metrics, are well established and have been successfully employed in numerous experiments in a variety of application domains [7], [8], [12], [21], [32], [33], [180], [184], [310]. However, the real-world application of more top-level predictors, such as scanpath-based task prediction, is still a very active research field. There is evidence that scanpath analysis enables reliable detection of specific significant patterns and tasks, e.g., reading or characteristic sequence of mirror checks [1], [11], [67], [188], [192]–[194], [196], [200], [203], [208]–[210]. Yet, less unequivocal predictions such as expertise and experience require precise demarcation of the application domain and the validity scope. Similar to the interpretation of the perception models, such models' application requires a robust and reliable detection of the system limits.

However, the presented models by far not limited to the automotive context. In fact, predicting the user's visual perception is highly relevant for AR and foveated rendering applications. These emerging HCI interfaces recently passed the prototype stage. Nevertheless, designing and developing minimally invasive yet efficient visual interfaces is still challenging. The presented user models enable realtime evaluation of the user's cognitive situation and visual awareness of presented scene objects. Novel HCI interfaces may incorporate such factors to provide adaptive and situation-aware displays individually attuned to the user's receptivity.

This thesis paved the way to gaze-driven vision-based HCI and HRI interfaces tailored to the user's situation awareness and cognitive visual receptivity.

Bibliography

The bibliography organizes into three lists of references. The first list *1st author publications* provides the thesis author's publications, wherein the thesis author appears as the first author in the conference, resp., journal article. These publications comprise the thesis author's major scientific contribution from 2017 to 2020. The second list *2nd author publications* provide thesis author's publications from 2016 to 2019, yet whereas the thesis author appears as the second author. In these publications, the thesis author significantly supported the project, e.g., by contributing to paper writing, technical support, evaluation, presentation at conferences. The *Other author publications* list provides the thesis author's publications from 2016 to 2020, wherein the thesis author provided minor contributions to the publication, e.g., proofreading, technical advice, provision, and adaptation of implementations. The *References* provides all publications wherein the thesis author was not involved, yet uses the published work as the fundament for own work, in particular the writing of this thesis.

1st author publications

[1] D. Geisler, N. Castner, G. Kasneci, and E. Kasneci, “A minhash approach for fast scanpath classification”, in *Eye Tracking Research & Applications (ETRA)*, 🏆 Best paper award, ACM, 2020.

[2] D. Geisler, D. Weber, N. Castner, and E. Kasneci, “Exploiting the gbvs for saliency aware gaze heatmaps”, in *Eye Tracking Research & Applications (ETRA)*, ACM, 2020.

[3] D. Geisler, P. Berens, and E. Kasneci, “A voyage through retinal signal processing: A bio-physiological driven perception model”, *Scientific Reports (SR)*, 2020, (UNDER REVIEW).

[4] D. Geisler, A. Duchowski, and E. Kasneci, “Predicting visual perceivability of scene objects through spatio-temporal modeling of retinal receptive fields”, *Neurocomputing (NEUROCOMP)*, 2020.

[5] D. Geisler, D. Fox, and E. Kasneci, “Real-time 3d glint detection in remote eye tracking based on bayesian inference”, in *International Conference on Robotics and Automation (ICRA)*, IEEE, 2018.

[6] D. Geisler, W. Fuhl, T. Santini, and E. Kasneci, “Saliency sandbox: Bottom-up saliency framework”, in *Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP)*, SCITEPRESS, 2017.

2nd author publications

[7] E. Bozkir, D. Geisler, and E. Kasneci, "Person independent, privacy preserving, and real time assessment of cognitive load using eye tracking in a virtual reality setup", in *Conference on Virtual Reality and 3D User Interfaces (VR)*, IEEE, 2019.

[8] E. Bozkir, D. Geisler, and E. Kasneci, "Assessment of driver attention during a safety critical situation in vr to generate vr-based training", in *Symposium on Applied Perception (SAP)*, ACM, 2019.

[9] W. Fuhl, D. Geisler, W. Rosenstiel, and E. Kasneci, "The applicability of cycle gans for pupil and eyelid segmentation, data generation and image refinement", in *International Conference on Computer Vision Workshops (ICCVW)*, IEEE, 2019.

[10] W. Fuhl, D. Geisler, T. Santini, T. Appel, W. Rosenstiel, and E. Kasneci, "Cbf:circular binary features for robust and real-time pupil center detection", in *Eye Tracking Research & Applications (ETRA)*, ACM, 2018.

[11] C. Braunagel, D. Geisler, W. Rosenstiel, and E. Kasneci, "Online recognition of driver-activity based on visual scanpath classification", *Intelligent Transportation Systems Magazine (ITSM)*, 2017.

[12] C. Braunagel, D. Geisler, W. Stolzmann, W. Rosenstiel, and E. Kasneci, "On the necessity of adaptive eye movement classification in conditionally automated driving scenarios", in *Eye Tracking Research & Applications (ETRA)*, ACM, 2016.

[13] W. Fuhl, D. Geisler, T. Santini, and E. Kasneci, "Evaluation of state-of-the-art pupil detection algorithms on remote eye images", in *Conference on Pervasive and Ubiquitous Computing (UBICOMP)*, ACM, 2016.

Other author publications

[14] B. Hosp, S. Eivazi, M. Maurer, W. Fuhl, D. Geisler, and E. Kasneci, "Remoteeye: An open-source high-speed remote eye tracker", *Behavior Research Methods*, 2020.

[15] W. Fuhl, E. Bozkir, B. Hosp, N. Castner, D. Geisler, T. Santini, and E. Kasneci, "Encodji: Encoding gaze data into emoji space for an amusing scanpath classification approach ;)" in *Eye Tracking Research & Applications*, ACM, 2019.

[16] K. Otto, N. Castner, D. Geisler, and E. Kasneci, "Development and evaluation of a gaze feedback system integrated into eyetrace", in *Eye Tracking Research & Applications (ETRA)*, ACM, 2018.

[17] T. Santini, W. Fuhl, D. Geisler, and E. Kasneci, "Eyerectoo: Open-source software for real-time pervasive head-mounted eye-tracking", in *Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP)*, SCITEPRESS, 2017.

[18] W. Fuhl, T. Santini, D. Geisler, T. Kübler, and E. Kasneci, "Eyelad: Remote eye tracking image labeling tool", in *Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP)*, SCITEPRESS, 2017.

[19] W. Fuhl, T. Santini, D. Geisler, T. Kübler, W. Rosenstiel, and E. Kasneci, "Eyes wide open? eyelid location and eye aperture estimation for pervasive eye tracking in real-world scenarios", in *Conference on Pervasive and Ubiquitous Computing (UBICOMP)*, ACM, 2016.

[20] M. Spika, J. Sonnenberg, H. Wendland, D. Geisler, T. Heinze, and J. Drüke, *Verfahren zum ermitteln einer objektwahrnehmungswahrscheinlichkeit, mit welcher ein fahrer eines kraftfahrzeugs objekte in einem umfeld des kraftfahrzeugs wahrnimmt, sowie steuervorrichtung für ein kraftfahrzeug und kraftfahrzeug*, Deutsches Patent und Markenamt (DPMA), Patent, DE 10 2018 200 980.5, 2019, updated 2020.

[277] J. Bickerdt, H. Wendland, D. Geisler, J. Sonnenberg, and E. Kasneci, "Beyond the tracked line of sight: Evaluation of visual perception parameters in a driving simulator setting", in *Journal of Eye Movement Research (JEMR)*, (UNDER REVIEW), Bielefeld University Library, 2020.

References

[21] M. W. Johns, A. Tucker, R. Chapman, K. Crowley, and N. Michael, "Monitoring eye and eyelid movements by infrared reflectance oculography to measure drowsiness in drivers", *Somnologie-Schlafforschung und Schlafmedizin*, 2007.

[22] Q. Wang, J. Yang, M. Ren, and Y. Zheng, "Driver fatigue detection: A survey", in *World Congress on Intelligent Control and Automation (WCICA)*, IEEE, 2006.

[23] F. Zhou, A. Alsaid, M. Blommer, R. Curry, R. Swaminathan, D. Kochhar, W. Talamonti, L. Tijerina, and B. Lei, "Driver fatigue transition prediction in highly automated driving using physiological features", *Expert Systems with Applications*, 2020.

[24] T. Poitschke, F. Laquai, and G. Rigoll, "Guiding a driver's visual attention using graphical and auditory animations", in *Engineering Psychology and Cognitive Ergonomics (EPCE)*, Springer, 2009.

[25] M. Bernhagen, P. Hein, A. Dettmann, and A. C. Bullinger, "Towards the development of a universal testing environment for attention guiding techniques", in *Applied Human Factors and Ergonomics (AHFE)*, Springer, 2019.

[26] Y. Higuchi, S. Inoue, H. Hamada, and T. Kumada, "Artificial optic flow guides visual attention in a driving scene", *Human factors*, 2020.

[27] X. Tian, Y. Niu, C. Xue, Y. Xie, B. Shi, B. Li, and L. Qiu, "Hud layout adaptive method for fighter based on flight mission", in *International Conference on Human Systems Engineering and Design (IHSED)*, Springer, 2019.

[28] M. Chi and L. Zhou, "Research on readability of adaptive foreground in dynamic background", in *Intelligent Human Systems Integration (IHSI)*, Springer, 2020.

[29] A. L. Yarbus, "Eye movements during perception of complex objects", in *Eye movements and vision*, Springer, 1967.

[30] J. Zagermann, U. Pfeil, and H. Reiterer, "Measuring cognitive load using eye tracking technology in visual computing", in *Beyond time and errors on novel evaluation methods for visualization (BELIV)*, ACM, 2016.

[31] R. Nomura and S. Maruno, "Rapid serial blinks: An index of temporally increased cognitive load", *Plos one*, 2019.

[32] T. Appel, N. Sevcenko, F. Wortha, K. Tsarava, K. Moeller, M. Ninaus, E. Kasneci, and P. Gerjets, "Predicting cognitive load in an emergency simulation based on behavioral and physiological measures", in *International Conference on Multimodal Interaction (ICMI)*, ACM, 2019.

[33] N. Castner, T. Appel, T. Eder, J. Richter, K. Scheiter, C. Keutel, F. Hüttig, A. Duchowski, and E. Kasneci, "Pupil diameter differentiates expertise in dental radiography visual search", *Plos one*,

[34] R. G. Macdonald and B. W. Tatler, "Gaze in a real-world social interaction: A dual eye-tracking study", *Journal of Experimental Psychology (JEP)*, 2018.

[35] S. L. Rogers, C. P. Speelman, O. Guidetti, and M. Longmuir, "Using dual eye tracking to uncover personal gaze patterns during social interaction", *Scientific reports (SR)*, 2018.

[36] F. van Ede, S. R. Chekroud, and A. C. Nobre, "Human gaze tracks attentional focusing in memorized visual space", *Human behaviour*, 2019.

[37] C. A. DiCesare, A. W. Kiefer, P. Nalepka, and G. D. Myer, "Quantification and analysis of saccadic and smooth pursuit eye movements and fixations to detect oculomotor deficits", *Behavior research methods (BRM)*, 2017.

[38] K. Harezlak and P. Kasprowski, "Application of eye tracking in medicine: A survey, research issues and challenges", *Computerized Medical Imaging and Graphics (CMIG)*, 2018.

[39] E. Kasneci, A. A. Black, and J. M. Wood, "Eye-tracking as a tool to evaluate functional ability in everyday tasks in glaucoma", *Journal of Ophthalmology (JOO)*, 2017.

[40] P. Kasprowski, K. Harezlak, and M. Niezabitowski, "Eye movement tracking as a new promising modality for human computer interaction", in *International Carpathian Control Conference (ICCC)*, IEEE, 2016.

[41] R. J. Jacob and K. S. Karn, "Eye tracking in human-computer interaction and usability research: Ready to deliver the promises", in *The mind's eye*, Elsevier, 2003.

[42] P. Berg and M. Scherg, "Dipole models of eye movements and blinks", *Electroencephalography and clinical Neurophysiology (ECN)*, 1991.

[43] M. E. Tagluk, N. Sezgin, and M. Akin, "Estimation of sleep stages by an artificial neural network employing eeg, emg and eog", *Journal of medical systems (JOMS)*, 2010.

[44] J. Schmidt, R. Laarousi, W. Stolzmann, and K. Karrer-Gauß, "Eye blink detection for different driver states in conditionally automated driving and manual driving using eog and a driver camera", *Behavior research methods (BRM)*, 2018.

[45] C. H. Morimoto and M. R. Mimica, "Eye gaze tracking techniques for interactive applications", *Computer vision and image understanding (CVIU)*, 2005.

[46] E. Whitmire, L. Trutoiu, R. Cavin, D. Perek, B. Scally, J. Phillips, and S. Patel, "Eyecontact: Scleral coil eye tracking for virtual reality", in *International Symposium on Wearable Computers (ISWC)*, ACM, 2016.

[51] D. Li, J. Babcock, and D. J. Parkhurst, "Openeyes: A low-cost head-mounted eye-tracking solution", in *Eye-Tracking Research & Applications (ETRA)*, ACM, 2006.

[52] S. Eivazi, T. C. Kübler, T. Santini, and E. Kasneci, "An inconspicuous and modular head-mounted eye tracker", in *Eye-Tracking Research & Applications (ETRA)*, ACM, 2018.

[53] M. Kassner, W. Patera, and A. Bulling, "Pupil: An open source platform for pervasive eye tracking and mobile gaze-based interaction", in *Adjunct Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, ACM, 2014.

[54] T. Santini, W. Fuhl, T. C. Kübler, and E. Kasneci, "Eyerec: An open-source data acquisition software for head-mounted eye-tracking.", in *Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP)*, SCITEPRESS, 2016.

[55] D. Li and D. Parkhurst, "Open-source software for real-time visible-spectrum eye tracking", in *Communication by Gaze Interaction (COGAIN)*, ACM, 2006.

[56] H. Sogo, "Gazeparser: An open-source and multiplatform library for low-cost eye tracking and analysis", *Behavior research methods (BRM)*, 2013.

[57] T. Santini, W. Fuhl, and E. Kasneci, "Pure: Robust pupil detection for real-time pervasive eye tracking", *Computer Vision and Image Understanding (CVIU)*, 2018.

[58] T. Santini, W. Fuhl, and E. Kasneci, "Purest: Robust pupil tracking for real-time pervasive eye tracking", in *Eye-Tracking Research & Applications (ETRA)*, ACM, 2018.

[59] D. Li, D. Winfield, and D. J. Parkhurst, "Starburst: A hybrid algorithm for video-based eye tracking combining feature-based and model-based approaches", in *Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2005.

[60] L. Świrski, A. Bulling, and N. Dodgson, "Robust real-time pupil tracking in highly off-axis images", in *Eye-Tracking Research & Applications (ETRA)*, ACM, 2012.

[61] A.-H. Javadi, Z. Hakimi, M. Barati, V. Walsh, and L. Tcheang, "Set: A pupil detection method using sinusoidal approximation", *Frontiers in neuroengineering*, 2015.

[62] W. Fuhl, T. Kübler, K. Sippel, W. Rosenstiel, and E. Kasneci, "Excuse: Robust pupil detection in real-world scenarios", in *Computer Analysis of Images and Patterns (CAIP)*, Springer, 2015.

[63] W. Fuhl, T. C. Santini, T. Kübler, and E. Kasneci, "Else: Ellipse selection for robust pupil detection in real-world environments", in *Eye-Tracking Research & Applications (ETRA)*, ACM, 2016.

[64] W. Fuhl, T. Santini, G. Kasneci, and E. Kasneci, "Pupilnet: Convolutional neural networks for robust pupil detection", *Computing Research Repository*, 2016.

[65] W. Fuhl, T. Santini, G. Kasneci, W. Rosenstiel, and E. Kasneci, "Pupilnet v2. 0: Convolutional neural networks for cpu based real time robust pupil detection", *Computing Research Repository*, 2017.

[66] M. Tonsen, X. Zhang, Y. Sugano, and A. Bulling, "Labelled pupils in the wild: A dataset for studying pupil detection in unconstrained environments", in *Eye-Tracking Research & Applications (ETRA)*, ACM, 2016.

[67] T. Kübler, "Algorithms for the comparison of visual scan patterns", PhD thesis, Eberhard Karls Universität Tübingen, 2016.

[68] T. C. Kübler, T. Rittig, E. Kasneci, J. Ungewiss, and C. Krauss, "Rendering refraction and reflection of eyeglasses for synthetic eye tracker images", in *Eye-Tracking Research & Applications (ETRA)*, ACM, 2016.

[69] T. Santini, W. Fuhl, and E. Kasneci, "Calibme: Fast and unsupervised eye tracker calibration for gaze-based pervasive human-computer interaction", in *Conference on Human Factors in Computing Systems (CHI)*, ACM, 2017.

[70] D. Mardanbegi and D. W. Hansen, "Parallax error in the monocular head-mounted eye trackers", in *Ubiquitous Computing (UBICOMP)*, ACM, 2012.

[71] D. C. Niehorster, T. Santini, R. S. Hessels, I. T. Hooge, E. Kasneci, and M. Nyström, "The impact of slippage on the data quality of head-worn eye trackers", *Behavior Research Methods (BRM)*, 2020.

[72] D. Mardanbegi, A. T. Kurauchi, and C. H. Morimoto, "An investigation of the distribution of gaze estimation errors in head mounted gaze trackers using polynomial functions", *Journal of Eye Movement Research (JEMR)*, 2018.

[73] T. Santini, D. Niehorster, and E. Kasneci, "Get a grip: Slippage-robust and glint-free gaze estimation for real-time pervasive head-mounted eye tracking", in *Eye-Tracking Research & Applications (ETRA)*, ACM, 2019.

[74] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion", *Pattern Recognition*, 2014.

[75] R. Safaee-Rad, I. Tchoukanov, K. C. Smith, and B. Benhabib, "Three-dimensional location estimation of circular features for machine vision", *Transactions on Robotics and Automation*, 1992.

[76] M. Toivanen, "An advanced kalman filter for gaze tracking signal", *Biomedical Signal Processing and Control*, 2016.

[77] L. Świrski and N. Dodgson, "A fully-automatic, temporal approach to single camera, glint-free 3d eye model fitting", *Journal of Eye Movement Research, Special Issue Pervasive eye tracking and mobile eye-based interaction (PETMEI)*, 2013.

[78] S. Eivazi, R. Bednarik, V. Leinonen, M. von und zu Fraunberg, and J. E. Jääskeläinen, "Embedding an eye tracker into a surgical microscope: Requirements, design, and implementation", *Sensors Journal*, 2015.

[79] H. Deubel and B. Bridgeman, "Fourth purkinje image signals reveal eye-lens deviations and retinal image distortions during saccades", *Vision research*, 1995.

[80] P. Majaranta and A. Bulling, "Eye tracking and eye-based human–computer interaction", in *Advances in physiological computing*, Springer, 2014.

[81] F. Vera-Olmos, E. Pardo, H. Melero, and N. Malpica, "Deepeye: Deep convolutional network for pupil detection in real environments", *Integrated Computer-Aided Engineering (ICAI)*, 2019.

[82] Y.-H. Yiu, M. Aboulatta, T. Raiser, L. Ophey, V. L. Flanagin, P. zu Eulenburg, and S.-A. Ahmadi, "Deepvog: Open-source pupil segmentation and gaze estimation in neuroscience using deep learning", *Journal of neuroscience methods (JONM)*, 2019.

[83] S. Eivazi, T. Santini, A. Keshavarzi, T. Kübler, and A. Mazzei, "Improving real-time cnn-based pupil detection through domain-specific data augmentation", in *Eye-Tracking Research & Applications (ETRA)*, ACM, 2019.

[84] A. Papoutsaki, P. Sangkloy, J. Laskey, N. Daskalova, J. Huang, and J. Hays, "Webgazer: Scalable webcam eye tracking using user interactions", in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2016.

[85] H. Chennamma and X. Yuan, "A survey on eye-gaze tracking techniques", *Computing Research Repository*, 2013.

[86] C. H. Morimoto, D. Koons, A. Amir, and M. Flickner, "Pupil detection and tracking using multiple light sources", *Image and vision computing (IVC)*, 2000.

[87] M. Bohme, A. Meyer, T. Martinetz, and E. Barth, "Remote eye tracking: State of the art and directions for future development", in *Communication by Gaze Interaction (COGAIN)*, Citeseer, 2006.

[88] M. Stengel, S. Grogorick, M. Eisemann, E. Eisemann, and M. A. Magnor, "An affordable solution for binocular eye tracking and calibration in head-mounted displays", in *International Conference on Multimedia (ICM)*, ACM, 2015.

[89] D. H. Yoo and M. J. Chung, "Non-intrusive eye gaze estimation without knowledge of eye pose", in *International Conference on Automatic Face and Gesture Recognition (ICAFGR)*, IEEE, 2004.

[90] Z. Zhu and Q. Ji, "Eye and gaze tracking for interactive graphic display", *Machine Vision and Applications (MVA)*, 2004.

[91] X. Yang, J. Sun, J. Liu, J. Chu, W. Liu, and Y. Gao, "A gaze tracking scheme for eye-based intelligent control", in *World Congress on Intelligent Control and Automation (WCICA)*, IEEE, 2010.

[92] Y. Ebisawa, S. Tsukahara, and D. Ishima, "Detection of feature points in video-based eye-gaze detection", in *Engineering in Medicine and Biology (EMBS)*, IEEE, 2002.

[93] A. Sharma and P. Abrol, "Glint detection and evaluation using edge detectors", *International Journal of Scientific and Technical Advancements (IJSTA)*, 2015.

[94] D. Droege and D. Paulus, "Pupil center detection in low resolution images", in *Eye-Tracking Research & Applications (ETRA)*, ACM, 2010.

[95] F. Timm and E. Barth, "Accurate eye centre localisation by means of gradients.", 2011.

[96] T. J. Atherton and D. J. Kerbyson, "Size invariant circle detection", *Image and Vision computing*, 1999.

[97] A. George and A. Routray, "Fast and accurate algorithm for eye localisation for gaze tracking in low-resolution images", *IET Computer Vision*, 2016.

[98] G. K. Aguirre, "A model of the entrance pupil of the human eye", *Scientific reports (SR)*, 2019.

[99] Y. Zhang, A. Bulling, and H. Gellersen, "Pupil-canthi-ratio: A calibration-free method for tracking horizontal gaze direction", in *Advanced Visual Interfaces (AVI)*, ACM, 2014.

[100] Y. Zhang, M. K. Chong, J. Müller, A. Bulling, and H. Gellersen, "Eye tracking for public displays in the wild", *Personal and Ubiquitous Computing (PUC)*, 2015.

[101] S.-y. Shin and S. Lee, "Correlation between car accident and car color for intelligent service", *Journal of Intelligence and Information Systems*, 2013.

[102] F. K. Alanazi, "Investigating the effect of vehicle color on crash risk", PhD thesis, University of Dayton, 2016.

[103] D. Eustace, F. K. Alanazi, and P. W. Hovey, "Investigation of the effect of vehicle color on safety", *Advances in transportation studies (ATS)*,

[104] A. Treisman, "Preattentive processing in vision", *Computer vision, graphics, and image processing*, 1985.

[105] T. Deng, K. Yang, Y. Li, and H. Yan, "Where does the driver look? top-down-based saliency detection in a traffic driving environment", *Intelligent Transportation Systems (TTS)*, 2016.

[106] C. Kanan, M. H. Tong, L. Zhang, and G. W. Cottrell, "Sun: Top-down saliency using natural statistics", *Visual cognition (VC)*, 2009.

[107] J. Yang and M.-H. Yang, "Top-down visual saliency via joint crf and dictionary learning", *Pattern analysis and machine intelligence (PAMI)*, 2016.

[108] J. Zhu, Y. Qiu, R. Zhang, J. Huang, and W. Zhang, "Top-down saliency detection via contextual pooling", *Signal Processing Systems (SPS)*, 2014.

[109] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset", *International Journal of Robotics Research (IJRR)*, 2013.

[110] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis", *IEEE Transactions on pattern analysis and machine intelligence*, 1998.

[111] D. Gao, V. Mahadevan, and N. Vasconcelos, "The discriminant center-surround hypothesis for bottom-up saliency", in *Neural information processing systems (NIPS)*, NIPS Foundation, 2008.

[112] V. Mahadevan and N. Vasconcelos, "Spatiotemporal saliency in dynamic scenes", *Pattern analysis and machine intelligence (PAMI)*, 2009.

[113] H. J. Seo and P. Milanfar, "Static and space-time visual saliency detection by self-resemblance", *Journal of vision (JOV)*, 2009.

[114] L. Zhang, M. H. Tong, T. K. Marks, H. Shan, and G. W. Cottrell, "Sun: A bayesian framework for saliency using natural statistics", *Journal of vision (JOV)*, 2008.

[115] C. Bak, A. Kocak, E. Erdem, and A. Erdem, "Spatio-temporal saliency networks for dynamic saliency prediction", *Transactions on Multimedia (TM)*, 2017.

[116] L. Jiang, M. Xu, and Z. Wang, "Predicting video saliency with object-to-motion cnn and two-layer convolutional lstm", *Computing Research Repository*, 2017.

[117] W. Wang, J. Shen, F. Guo, M.-M. Cheng, and A. Borji, "Revisiting video saliency: A large-scale benchmark and a new model", in *Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2018.

[118] W. Wang, J. Shen, and L. Shao, "Video salient object detection via fully convolutional networks", *Transactions on Image Processing (TIP)*, 2017.

[119] M. Yanoff, B. S. Fine, and J. D. M. Gass, *Ocular pathology*. Mosby-Wolfe London, 1996.

[120] C. Behrens, T. Schubert, S. Haverkamp, T. Euler, and P. Berens, "Connectivity map of bipolar cells and photoreceptors in the mouse retina", *Elife*, 2016.

[121] C. A. Curcio, K. R. Sloan, R. E. Kalina, and A. E. Hendrickson, "Human photoreceptor topography", *Journal of comparative neurology*, 1990.

[122] C. A. Curcio, K. A. Allen, K. R. Sloan, C. L. Lerea, J. B. Hurley, I. B. Klock, and A. H. Milam, "Distribution and morphology of human cone photoreceptors stained with anti-blue opsin", *Journal of Comparative Neurology*, 1991.

[123] J. K. Bowmaker and H. Dartnall, "Visual pigments of rods and cones in a human retina.", *The Journal of physiology*, 1980.

[124] G. Wyszecki and W. S. Stiles, *Color science*. Wiley New York, 1982.

[125] D. Purves, G. Augustine, D. Fitzpatrick, L. Katz, A. LaMantia, J. McNamara, and S. Williams, *Neuroscience 2nd Edition. Sunderland (MA) Sinauer Associates*. Inc, 2001.

[126] Á. Szél, P. Röhlich, A. R. Caffé, and T. Van Veen, "Distribution of cone photoreceptors in the mammalian retina", *Microscopy research and technique*, 1996.

[127] H. Kolb, E. Fernandez, J. Schouten, P. Ahnelt, K. A. Linberg, and S. K. Fisher, "Are there three types of horizontal cell in the human retina?", *Journal of Comparative Neurology*, 1994.

[128] S. L. Mills and S. C. Massey, "Distribution and coverage of a-and b-type horizontal cells stained with neurobiotin in the rabbit retina", *Visual neuroscience*, 1994.

[129] I. Perlman, H. Kolb, and R. Nelson, "S-potentials and horizontal cells", *Webvision*, 1995.

[130] I. Perlman, E. Weiner, and H. Kolb, "Retinal horizontal cells", *Encyclopedia of Neuroscience*, 2009.

[131] T. Euler, S. Haverkamp, T. Schubert, and T. Baden, "Retinal bipolar cells: Elementary building blocks of vision", *Nature Reviews Neuroscience*, 2014.

[132] H. Kolb, E. Fernandez, and R. Nelson, "Bipolar cell pathways in the vertebrate retina – the organization of the retina and visual system", *Webvision*, 1995.

[133] Y. Tsukamoto and N. Omi, "On bipolar cells in macaque retina: Type-specific synaptic connectivity with special reference to off counterparts", *Frontiers in neuroanatomy*, 2016.

[134] D. M. Dacey, J. D. Crook, and O. S. Packer, "Distinct synaptic mechanisms create parallel s-on and s-off color opponent pathways in the primate retina", *Visual Neuroscience*, 2014.

[135] Y. Tsukamoto and N. Omi, "Off bipolar cells in macaque retina: Type-specific connectivity in the outer and inner synaptic layers", *Frontiers in neuroanatomy*, 2015.

[136] H. Kolb, "Circuitry for rod signals through the retina", *Webvision*, 2009.

[137] U. Grünert and P. R. Martin, "Cell types and cell circuits in human and non-human primate retina", *Progress in Retinal and Eye Research (PRER)*, 2020.

[138] R. H. Masland, "The tasks of amacrine cells", *Visual neuroscience*, 2012.

[139] H. Kolb, "Roles of amacrine cells", *Webvision*, 2007.

[140] D. I. Vaney, "The mosaic of amacrine cells in the mammalian retina", *Progress in retinal research*, 1990.

[141] R. Balasubramanian and L. Gan, "Development of retinal amacrine cells and their dendritic stratification", *Current ophthalmology reports*, 2014.

[142] W. Taylor and R. Smith, "The role of starburst amacrine cells in visual signal processing", *Visual neuroscience*, 2012.

[143] R. E. Marc, J. R. Anderson, B. W. Jones, C. L. Sigulinsky, and J. S. Lauritzen, "The aii amacrine cell connectome: A dense network hub", *Frontiers in neural circuits*, 2014.

[144] D. M. Dacey, B. B. Peterson, F. R. Robinson, and P. D. Gamlin, "Fireworks in the primate retina: In vitro photodynamics reveals diverse lgn-projecting ganglion cell types", *Neuron*, 2003.

[145] B. B. Lee, "Receptive field structure in the primate retina", *Vision research*, 1996.

[146] J. R. Sanes and R. H. Masland, "The types of retinal ganglion cells: Current status and implications for neuronal classification", *Annual review of neuroscience*, 2015.

[147] E. Kaplan and R. M. Shapley, "The primate retina contains two types of ganglion cells, with high and low contrast sensitivity", *Proceedings of the National Academy of Sciences*, 1986.

[148] T. Baden, P. Berens, K. Franke, M. R. Rosón, M. Bethge, and T. Euler, "The functional diversity of retinal ganglion cells in the mouse", *Nature*, 2016.

[149] N. Tian, "Development of retinal ganglion cell dendritic structure and synaptic connections", *Webvision*, 2012.

[150] D. M. Dacey and M. R. Petersen, "Dendritic field size and morphology of midget and parasol ganglion cells of the human retina.", *Proceedings of the National Academy of sciences*, 1992.

[151] H. Wässle, U. Grünert, J. Röhrenbeck, and B. B. Boycott, "Retinal ganglion cell density and cortical magnification factor in the primate", *Vision research*, 1990.

[152] C. A. Curcio and K. A. Allen, "Topography of ganglion cells in human retina", *Journal of comparative Neurology*, 1990.

[153] L. Silveira and V. Perry, "The topography of magnocellular projecting ganglion cells (m-ganglion cells) in the primate retina", *Neuroscience*, 1991.

[154] R. W. Rodieck, "Quantitative analysis of cat retinal ganglion cell response to visual stimuli", *Vision research*, 1965.

[155] C. Bader and D. Bertrand, "Effect of changes in intra-and extracellular sodium on the inward (anomalous) rectification in salamander photoreceptors.", *The Journal of physiology*, 1984.

[156] D. M. Schneeweis and J. L. Schnapf, "The photovoltage of macaque cone photoreceptors: Adaptation, noise, and kinetics", *Journal of Neuroscience*, 1999.

[157] G. C. Demontis, B. Longoni, U. Barcaro, and L. Cervetto, "Properties and functional roles of hyperpolarization-gated currents in guinea-pig retinal rods", *The Journal of Physiology*, 1999.

[158] W. Cobbs and E. Pugh, "Kinetics and components of the flash photocurrent of isolated retinal rods of the larval salamander, ambystoma tigrinum.", *The Journal of physiology*, 1987.

[159] R. H. Masland, "The fundamental plan of the retina", *Nature neuroscience*, 2001.

[160] B. Cleland, M. Dubin, and W. Levick, "Sustained and transient neurones in the cat's retina and lateral geniculate nucleus", *The Journal of physiology*, 1971.

[161] G. B. Awatramani and M. M. Slaughter, "Origin of transient and sustained responses in ganglion cells of the retina", *Journal of Neuroscience*, 2000.

[162] X. Zhao, A. N. Reifler, M. M. Schroeder, E. R. Jaeckel, A. P. Chervenak, and K. Y. Wong, "Mechanisms creating transient and sustained photoresponses in mammalian retinal ganglion cells", *The Journal of general physiology*, 2017.

[163] M. H. Hennig, K. Funke, and F. Wörgötter, "The influence of different retinal subcircuits on the nonlinearity of ganglion cell behavior", *Journal of Neuroscience*, 2002.

[164] S. Nirenberg, S. M. Carcieri, A. L. Jacobs, and P. E. Latham, "Retinal ganglion cells act largely as independent encoders", *Nature*, 2001.

[165] S. Nirenberg and P. E. Latham, "Population coding in the retina", *Current Opinion in Neurobiology*, 1998.

[166] E. Fernandez, J.-M. Ferrandez, J. Ammermüller, and R. A. Normann, "Population coding in spike trains of simultaneously recorded retinal ganglion cells1", *Brain research*, 2000.

[167] A. Wohrer and P. Kornprobst, "Virtual retina: A biological retina model and simulator, with contrast gain control", *Journal of computational neuroscience*, 2009.

[168] C. Bradley, J. Abrams, and W. S. Geisler, "Retina-v1 model of detectability across the visual field", *Journal of vision*, 2014.

[169] A. T. Duchowski, "Eye tracking methodology", *Theory and practice*, 2007.

[170] K. Holmqvist, M. Nyström, R. Andersson, R. Dewhurst, H. Jarodzka, and J. Van de Weijer, *Eye tracking: A comprehensive guide to methods and measures*. OUP Oxford, 2011.

[171] A. Kennedy, *Book Review: Eye Tracking: A Comprehensive Guide to Methods and Measures*. SAGE, 2016.

[172] D. D. Salvucci and J. H. Goldberg, "Identifying fixations and saccades in eye-tracking protocols", in *Eye-Tracking Research & Applications (ETRA)*, ACM, 2000.

[173] O. V. Komogortsev and A. Karpov, "Automated classification and scoring of smooth pursuit eye movements in the presence of fixations and saccades", *Behavior research methods (BRM)*, 2013.

[174] E. Kasneci, G. Kasneci, T. C. Kübler, and W. Rosenstiel, "Online recognition of fixations, saccades, and smooth pursuits for automated analysis of traffic hazard perception", in *Artificial Neural Networks (ANN)*, Springer, 2015.

[175] E. Kasneci, G. Kasneci, T. C. Kübler, and W. Rosenstiel, "The applicability of probabilistic methods to the online recognition of fixations and saccades in dynamic scenes", in *Eye-Tracking Research & Applications (ETRA)*, ACM, 2014.

[176] D. D. Salvucci, *Mapping eye movements to cognitive processes*. Carnegie Mellon University Pittsburgh, 1999.

[177] D. D. Salvucci and J. R Anderson, "Tracing eye movement protocols with cognitive process models", 1998.

[178] K. Wegner-Clemens, J. Rennig, J. F. Magnotti, and M. S. Beauchamp, "Using principal component analysis to characterize eye movement fixation patterns during face viewing", *Journal of vision (JOV)*, 2019.

[179] L. Larsson, M. Nyström, R. Andersson, and M. Stridh, "Detection of fixations and smooth pursuit movements in high-speed eye-tracking data", *Biomedical Signal Processing and Control (BSPC)*, 2015.

[180] S. Mathôt, "Pupillometry: Psychology, physiology, and function", *Journal of Cognition (JOC)*, 2018.

[181] A. A. Black, J. M. Wood, M. J. Collins, and G. Isoardi, "Variations in pupil size and light levels while driving at night", in *ARVO Meeting Abstracts*, QUT, 2019.

[182] W. Fuhl, T. Santini, and E. Kasneci, "Fast and robust eyelid outline and aperture detection in real-world scenarios", in *Winter Conference on Applications of Computer Vision (WACV)*, IEEE, 2017.

[183] W. Fuhl, W. Rosenstiel, and E. Kasneci, "500,000 images closer to eyelid and pupil segmentation", in *Computer Analysis of Images and Patterns (CAIP)*, Springer, 2019.

[184] F. Yang, X. Yu, J. Huang, P. Yang, and D. Metaxas, "Robust eyelid tracking for fatigue detection", in *International Conference on Image Processing (ICIP)*, IEEE, 2012.

[185] S. Rostaminia, A. Mayberry, D. Ganesan, B. Marlin, and J. Gummeson, "Ilid: Eyewear solution for low-power fatigue and drowsiness monitoring", in *Eye-Tracking Research & Applications (ETRA)*, ACM, 2019.

[186] S. O. Kobald, E. Wascher, H. Heppner, and S. Getzmann, "Eye blinks are related to auditory information processing: Evidence from a complex speech perception task", *Psychological research*, 2019.

[187] J. Renker, M. Kreutzfeldt, and G. Rinkenauer, "Eye blinks describing the state of the learner under uncertainty", in *International Conference on Human-Computer Interaction*, Springer, 2019.

[188] N. C. Anderson, F. Anderson, A. Kingstone, and W. F. Bischof, "A comparison of scanpath comparison methods", *Behavior research methods (BRM)*, 2015.

[189] C. Berger, M. Winkels, A. Lischke, and J. Höppner, "Gazealyze: A matlab toolbox for the analysis of eye movement data", *Behavior research methods (BRM)*, 2012.

[190] T. Kübler, W. Fuhl, R. Rosenberg, W. Rosenstiel, and E. Kasneci, "Novel methods for analysis and visualization of saccade trajectories", in *European Conference on Computer Vision (ECCV)*, Springer, 2016.

[191] M. Raschke, D. Herr, T. Blascheck, T. Ertl, M. Burch, S. Willmann, and M. Schrauf, "A visual approach for scan path comparison", in *Eye-Tracking Research & Applications (ETRA)*, ACM, 2014.

[192] R. Dewhurst, M. Nyström, H. Jarodzka, T. Foulsham, R. Johansson, and K. Holmqvist, "It depends on how you look at it: Scanpath comparison in multiple dimensions with multimatch, a vector-based approach", *Behavior research methods*, 2012.

[193] T. Foulsham, R. Dewhurst, M. Nyström, H. Jarodzka, R. Johansson, G. Underwood, and K. Holmqvist, "Comparing scanpaths during scene encoding and recognition: A multi-dimensional approach", *Journal of eye movement research (JEMR)*, 2012.

[194] H. Jarodzka, K. Holmqvist, and M. Nyström, "A vector-based, multidimensional scanpath similarity measure", in *Eye-Tracking Research & Applications (ETRA)*, ACM, 2010.

[195] S. Mathôt, F. Cristino, I. D. Gilchrist, and J. Theeuwes, "A simple way to estimate similarity between pairs of eye movement sequences", *Journal of eye movement research (JEMR)*, 2012.

[196] R. M. Foerster and W. X. Schneider, “Functionally sequenced scanpath similarity method (funcsim): Comparing and evaluating scanpath similarity based on a task’s inherent sequence of functional (action) units”, *Journal of eye movement research (JEMR)*, 2014.

[197] T. C. Kübler, E. Kasneci, and W. Rosenstiel, “Subsmatch: Scanpath similarity in dynamic scenes based on subsequence frequencies”, in *Eye-Tracking Research & Applications (ETRA)*, ACM, 2014.

[198] W. Fuhl, T. C. Kübler, H. Brinkmann, R. Rosenberg, W. Rosenstiel, and E. Kasneci, “Region of interest generation algorithms for eye tracking data.”, in *Eye Tracking and Visualization (ETVIS)*, ACM, 2018.

[199] W. Fuhl, T. C. Kübler, T. Santini, and E. Kasneci, “Automatic generation of saliency-based areas of interest for the visualization and analysis of eye-tracking data.”, in *International Symposium on Vision, Modeling, and Visualization (VMV)*, ACM, 2018.

[200] J. Heminghous and A. T. Duchowski, “Icomp: A tool for scanpath visualization and comparison.”, in *SIGGRAPH Research Posters*, ACM, 2006.

[201] S. B. Needleman and C. D. Wunsch, “A general method applicable to the search for similarities in the amino acid sequence of two proteins”, *Journal of molecular biology (J. Mol. Biol)*, 1970.

[202] V. I. Levenshtein, “Binary codes capable of correcting deletions, insertions, and reversals”, in *Soviet physics doklady*, MAIK Nauka/Interperiodica, 1966.

[203] N. Castner, E. Kasneci, T. Kübler, K. Scheiter, J. Richter, T. Eder, F. Hüttig, and C. Keutel, “Scanpath comparison in medical image reading skills of dental students: Distinguishing stages of expertise development”, in *Eye-Tracking Research & Applications (ETRA)*, ACM, 2018.

[204] F. Cristino, S. Mathôt, J. Theeuwes, and I. D. Gilchrist, “Scanmatch: A novel method for comparing fixation sequences”, *Behavior research methods*, 2010.

[205] F. Deitelhoff, A. Harrer, and A. Kienle, “The influence of different aoi models in source code comprehension analysis”, in *Proceedings of the 6th International Workshop on Eye Movements in Programming*, IEEE Press, 2019.

[206] T. Busjahn, R. Bednarik, A. Begel, M. Crosby, J. H. Paterson, C. Schulte, B. Sharif, and S. Tamm, “Eye movements in code reading: Relaxing the linear order”, in *Program Comprehension (ICPC), 2015 IEEE 23rd International Conference on*, IEEE, 2015.

[207] R.-F. Day, “Examining the validity of the needleman–wunsch algorithm in identifying decision strategy with eye-movement data”, *Decision Support Systems*, 2010.

[208] T. C. Kübler, C. Rothe, U. Schiefer, W. Rosenstiel, and E. Kasneci, “Subsmatch 2.0: Scanpath comparison and classification based on subsequence frequencies”, *Behavior research methods*, 2017.

[209] T. C. Kübler and E. Kasneci, "Automated comparison of scanpaths in dynamic scenes", in *SAGA-International Workshop on Solutions for Automatic Gaze Data Analysis: Proceedings*, Bielefeld University Library, 2015.

[210] T. Kübler, S. Eivazi, and E. Kasneci, "Automated visual scanpath analysis reveals the expertise level of micro-neurosurgeons", in *MICCAI workshop on interventional microscopy*, Springer, 2015.

[211] M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua, "Fast keypoint recognition using random ferns", *Pattern analysis and machine intelligence (PAMI)*, 2009.

[212] M. Villamizar, F. Moreno-Noguer, J. Andrade-Cetto, and A. Sanfeliu, "Efficient rotation invariant object detection using boosted random ferns", in *Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2010.

[213] M. Villamizar, J. Andrade-Cetto, A. Sanfeliu, and F. Moreno-Noguer, "Boosted random ferns for object detection", *Pattern analysis and machine intelligence (PAMI)*, 2017.

[214] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets", in *Neural information processing systems (NIPS)*, NIPS Foundation, 2014.

[215] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks", in *International conference on computer vision (ICCV)*, IEEE, 2017.

[216] K. Cheng, R. Tahir, L. K. Eric, and M. Li, "An analysis of generative adversarial networks and variants for image synthesis on mnist dataset", *Multimedia Tools and Applications (MTA)*, 2020.

[217] R. Wan, B. Shi, L.-Y. Duan, A.-H. Tan, and A. C. Kot, "Benchmarking single-image reflection removal algorithms", in *International Conference on Computer Vision (ICCV)*, IEEE, 2017.

[218] W. Fuhl, N. Castner, L. Zhuang, M. Holzer, W. Rosenstiel, and E. Kasneci, "Mam: Transfer learning for fully automatic video annotation and specialized detector creation", in *European Conference on Computer Vision (ECCV)*, Springer, 2018.

[219] V. Kazemi and J. Sullivan, "One millisecond face alignment with an ensemble of regression trees", in *Computer vision and pattern recognition (CVPR)*, IEEE, 2014.

[220] D. E. King, "Dlib-ml: A machine learning toolkit", *Journal of Machine Learning Research (JMLR)*, 2009.

[221] W. Fuhl, T. Santini, and E. Kasneci, "Fast and robust eyelid outline and aperture detection in real-world scenarios", in *Winter Conference on Applications of Computer Vision (WACV)*, IEEE, 2017.

[222] C. Hennessey, B. Noureddin, and P. Lawrence, "A single camera eye-gaze tracking system with free head motion", in *Eye-Tracking Research & Applications (ETRA)*, ACM, 2006.

[223] A. Sharma and P. Abrol, "Direction estimation model for gaze controlled systems", *Journal of Eye Movement Research (JEMR)*, 2016.

[224] R. Hartley and P. Sturm, "Triangulation", in *Computer analysis of images and patterns (CAIP)*, Springer, 1995.

[225] P. Lindstrom, "Triangulation made easy", in *Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2010.

[226] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision Second Edition*. Cambridge University Press, 2000.

[227] N. A. Dodgson *et al.*, "Variation and extrema of human interpupillary distance", in *Stereoscopic Displays and Virtual Reality Systems XI*, International Society for Optics and Photonics, 2004.

[228] P. Diaconis and D. Ylvisaker, "Conjugate priors for exponential families", *The Annals of statistics*, 1979.

[229] J. Joyce, "Bayes' theorem", in *The Stanford Encyclopedia of Philosophy*, Metaphysics Research Lab, Stanford University, 2016.

[230] L. Bottou, *Neural networks: Tricks of the trade*. Springer, 2012.

[231] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding", 2014.

[232] T. Baltrusaitis, P. Robinson, and L.-P. Morency, "Constrained local neural fields for robust facial landmark detection in the wild", in *International Conference on Computer Vision (ICCV)*, IEEE, 2013.

[233] M. Rezaei and R. Klette, "Adaptive haar-like classifier for eye status detection under non-ideal lighting conditions", in *Conference on Image and Vision Computing New Zealand (CIVCNZ)*, ACM, 2012.

[234] O. Jesorsky, K. J. Kirchberg, and R. W. Frischholz, "Robust face detection using the hausdorff distance", in *Audio- and Video-based Biometric Person Authentication (AVBPA)*, Springer, 2001.

[235] A. Villanueva, V. Ponz, L. Sesma-Sanchez, M. Ariz, S. Porta, and R. Cabeza, "Hybrid method based on topography for robust detection of iris center and eye corners", *Multimedia Computing, Communications, and Applications (MCCA)*, 2013.

[236] F. Wallhoff, "Fgnet-facial expression and emotion database", 2004.

[237] C. Tomasi and T. Kanade, *Detection and tracking of point features*. School of Computer Science, Carnegie Mellon Univ. Pittsburgh, 1991.

[238] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features", in *Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2001.

[239] G. Bradski, "Opencv library", *Dr. Dobb's Journal of Software Tools*, 2000.

[240] J. Harel, C. Koch, and P. Perona, "Graph-based visual saliency", in *Advances in neural information processing systems (NIPS)*, NIPS Foundation, 2007.

[241] X. Hou and L. Zhang, "Saliency detection: A spectral residual approach", in *Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2007.

[242] R. Achanta, S. Hemami, F. Estrada, and S. Susstrunk, "Frequency-tuned salient region detection", in *Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2009.

[243] L. Itti and C. Koch, "A saliency-based search mechanism for overt and covert shifts of visual attention", *Vision Research (VR)*, 2000.

[244] J. Zhang and S. Sclaroff, "Saliency detection: A boolean map approach", in *International Conference on Computer Vision (ICCV)*, IEEE, 2013.

[245] ITU, *Studio encoding parameters of digital television for standard 4:3 and wide screen 16:9 aspect ratios*, ITU, 2011.

[246] A. M. Derrington, J. Krauskopf, and P. Lennie, "Chromatic mechanisms in lateral geniculate nucleus of macaque.", *The Journal of Physiology*, 1984.

[247] A. M. Treisman and G. Gelade, "A feature-integration theory of attention", *Cognitive psychology*, 1980.

[249] Z. Zivkovic, "Improved adaptive gaussian mixture model for background subtraction", in *International Conference on Pattern Recognition (ICPR)*, IEEE, 2004.

[250] P. KaewTraKulPong and R. Bowden, "An improved adaptive background mixture model for real-time tracking with shadow detection", in *Video-based surveillance systems*, Springer, 2002.

[251] A. B. Godbehere, A. Matsukawa, and K. Goldberg, "Visual tracking of human visitors under variable-lighting conditions for a responsive audio art installation", in *American Control Conference (ACC)*, IEEE, 2012.

[252] B. Schauerte and R. Stiefelhagen, "Quaternion-based spectral saliency detection for eye fixation prediction", in *European Conference on Computer Vision (ECCV)*, Springer, 2012.

[253] D. R. van Renswoude, I. Visser, M. E. Raijmakers, T. Tsang, and S. P. Johnson, "Real-world scene perception in infants: What factors guide attention allocation?", *Infancy*, 2019.

[254] I. Kotseruba and J. K. Tsotsos, "40 years of cognitive architectures: Core cognitive abilities and practical applications", *Artificial Intelligence Review*, 2018.

[255] R. Yuster and U. Zwick, "Fast sparse matrix multiplication", *ACM Transactions On Algorithms (TALG)*, 2005.

[256] T. Roosendaal, "Big buck bunny", in *ACM SIGGRAPH ASIA 2008 Computer Animation Festival*, Association for Computing Machinery, 2008.

[257] A. Olsen, "The tobii i-vt fixation filter", *Tobii Technology*, 2012.

[258] N. T. Choplin and R. P. Edwards, *Visual fields*. Slack Incorporated, 1998.

[259] M. Lavidor and V. Walsh, "The nature of foveal representation", *Nature Reviews Neuroscience*, 2004.

[260] J. M. Van Buren, *The retinal ganglion cell layer: a physiological-anatomical correlation in man and primates of the normal topographical anatomy of the retinal ganglion cell layer and its alterations with lesions of the visual pathways*. Charles C. Thomas, 1963.

[261] R. G. Michels, C. P. Wilkinson, T. A. Rice, and T. C. Hengst, *Retinal detachment*. Mosby St Louis, 1990.

[262] H. Kolb, E. Fernandez, and R. Nelson, "Facts and figures concerning the human retina – the organization of the retina and visual system", *Webvision*, 1995.

[263] T. D. Lamb, "Why rods and cones?", *PubMed*, 2015.

[264] T. Hansen, L. Pracejus, and K. R. Gegenfurtner, "Color perception in the intermediate periphery of the visual field", *Journal of vision (JOV)*, 2009.

[265] C. A. Curcio, K. R. Sloan, O. Packer, A. E. Hendrickson, and R. E. Kalina, "Distribution of cones in human and monkey retina: Individual variability and radial asymmetry", *Science*, 1987.

[266] S. J. Williamson and H. Z. Cummins, "Light and color in nature and art", *American Journal of Physics*, 1984.

[267] H. Strasburger, I. Rentschler, and M. Jüttner, "Peripheral vision and pattern recognition: A review", *Journal of vision (JOV)*, 2011.

[268] H. Kolb, "How the retina works: Much of the construction of an image takes place in the retina itself through the use of specialized neural circuits", *American scientist*, 2003.

[269] H. Kolb, "Photoreceptors", *Webvision*, 2013.

[270] S. Borer and S. Süsstrunk, "Opponent color space motivated by retinal processing", in *Conference on colour in graphics, imaging, and vision*, 2002.

[272] M. Lankheet, J. Molenaar, and W. Van De Grind, "The spike generating mechanism of cat retinal ganglion cells", *Vision research*, 1989.

[273] J. Fohlmeister, P. Coleman, and R. Miller, "Modeling the repetitive firing of retinal ganglion cells", *Brain research*, 1990.

[274] J. Fohlmeister and R. Miller, "Impulse encoding mechanisms of ganglion cells in the tiger salamander retina", *Journal of neurophysiology*, 1997.

[275] M. A. Sikora, J. Gottesman, and R. F. Miller, "A computational model of the ribbon synapse", *Journal of neuroscience methods*, 2005.

[276] L. Wang, P.-J. Liang, P.-M. Zhang, and Y.-h. Qiu, "Ionic mechanisms underlying tonic and phasic firing behaviors in retinal ganglion cells", *channels*, 2014.

[278] M. H. Turner, G. W. Schwartz, and F. Rieke, "Receptive field center-surround interactions mediate context-dependent spatial contrast encoding in the retina", *bioRxiv*, 2018.

[279] C. Enroth-Cugell and P. Lennie, “The control of retinal ganglion cell discharge by receptive field surrounds.”, *The Journal of physiology*, 1975.

[280] A. Ransom-Hogg and L. Spillmann, “Perceptive field size in fovea and periphery of the light-and dark-adapted retina”, *Vision Research*, 1980.

[281] K. A. Zaghloul, K. Boahen, and J. B. Demb, “Different circuits for on and off retinal ganglion cells cause different contrast sensitivities”, *Journal of Neuroscience*, 2003.

[282] M. D. Fairchild, *Color appearance models*. John Wiley & Sons, 2013.

[283] E. Reinhard, M. Adhikhmin, B. Gooch, and P. Shirley, “Color transfer between images”, *IEEE Computer graphics and applications*, 2001.

[284] B. Crawford, “The scotopic visibility function”, *Proceedings of the Physical Society. Section B*, 1949.

[285] G. Wald, “Human vision and the spectrum.”, *Science*, 1945.

[286] D. B. Judd, “Color in business, science and industry.”, *Journal of the Society of Motion Picture and Television Engineers*, 1952.

[287] W. Stiles and J. Burch, “Interim report to the commission internationale de l’eclairage, zurich, 1955, on the national physical laboratory’s investigation of colour-matching (1955)”, *Optica Acta: International Journal of Optics*, 1955.

[288] A. Stockman and L. T. Sharpe, “The spectral sensitivities of the middle-and long-wavelength-sensitive cones derived from measurements in observers of known genotype”, *Vision research*, 2000.

[289] H. Kolb, “Midget pathways of the primate retina underlie resolution and red green color opponency”, *Webvision*, 1995.

[290] H. Kolb, P. Goede, S. Roberts, R. McDermott, and P. Gouras, “Uniqueness of the s-cone pedicle in the human retina and consequences for color processing”, *Journal of Comparative Neurology*, 1997.

[291] R. Jacoby, D. Stafford, N. Kouyama, and D. Marshak, “Synaptic inputs to on parasol ganglion cells in the primate retina”, *Journal of Neuroscience*, 1996.

[292] V. J. Volbrecht, E. E. Shrago, B. E. Schefrin, and J. S. Werner, “Spatial summation in human cone mechanisms from 0°to 20°in the superior retina”, *JOSA A*, 2000.

[294] H. Kolb and L. Lipetz, “The anatomical basis for colour vision in the vertebrate retina”, *Vision and visual dysfunction*, 1991.

[295] R. E. Marc, “Chromatic organization of the retina”, *Cell biology of the eye*, 1982.

[296] M. Barz, A. Bulling, and F. Daiber, “Computational modelling and prediction of gaze estimation error for head-mounted eye trackers”, *DFKI ResearchReports*, 2015.

[297] A. Bykowski and S. Kupiński, “Automatic mapping of gaze position coordinates of eye-tracking glasses video on a common static reference image”, in *Eye-Tracking Research & Applications (ETRA)*, ACM, 2018.

[298] G. H. Golub and C. Reinsch, "Singular value decomposition and least squares solutions", *Numerische mathematik*, 1970.

[299] L. Breiman, *Classification and regression trees*. Routledge, 2017.

[300] L. Breiman, "Bagging predictors", *Machine learning*, 1996.

[301] R. E. Schapire, Y. Freund, P. Bartlett, W. S. Lee, *et al.*, "Boosting the margin: A new explanation for the effectiveness of voting methods", *The annals of statistics*, 1998.

[302] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "Rusboost: Improving classification performance when training data is skewed", in *2008 19th International Conference on Pattern Recognition*, IEEE, 2008.

[303] I. Barandiaran, "The random subspace method for constructing decision forests", *IEEE Trans. Pattern Anal. Mach. Intell*, 1998.

[304] A. J. Dobson and A. G. Barnett, *An introduction to generalized linear models*. CRC press, 2018.

[305] J. Shawe-Taylor and N. Cristianini, *An introduction to support vector machines and other kernel-based learning methods*. Cambridge University Press Cambridge, 2000.

[306] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression", *The American Statistician*, 1992.

[307] P. Vetter and A. Newen, "Varieties of cognitive penetration in visual perception", *Consciousness and cognition*, 2014.

[308] C. Ellis, "The pupillary light reflex in normal subjects.", *British Journal of Ophthalmology*, 1981.

[309] K. Preuschoff, B. M. t Hart, and W. Einhauser, "Pupil dilation signals surprise: Evidence for noradrenaline's role in decision making", *Frontiers in neuroscience*, 2011.

[310] B. C. Goldwater, "Psychological significance of pupillary movements.", *Psychological bulletin*, 1972.

[311] M. B. Bender and M. A. Kennard, "The fright reaction after section of the facial, trigeminal and cervical sympathetic nerves", *Journal of Neurophysiology*, 1938.

[312] S. Yrttiaho, D. Niehaus, E. Thomas, and J. M. Leppänen, "Mothers' pupillary responses to infant facial expressions", *Behavioral and Brain Functions*, 2017.

[313] M. De Rivecourt, M. Kuperus, W. Post, and L. Mulder, "Cardiovascular and eye activity measures as indices for momentary changes in mental effort during simulated flight", *Ergonomics*, 2008.

[314] K. F. Van Orden, W. Limbert, S. Makeig, and T.-P. Jung, "Eye activity correlates of workload during a visuospatial memory task", *Human factors*, 2001.

[315] A. Savitzky and M. J. Golay, "Smoothing and differentiation of data by simplified least squares procedures.", *Analytical chemistry*, 1964.

[316] V. Svetnik, A. Liaw, C. Tong, J. C. Culberson, R. P. Sheridan, and B. P. Feuston, "Random forest: A classification and regression tool for compound classification and qsar modeling", *Journal of chemical information and computer sciences*, 2003.

[317] E. Tafaj, T. C. Kübler, G. Kasneci, W. Rosenstiel, and M. Bogdan, "Online classification of eye tracking data for automated analysis of traffic hazard perception", in *International Conference on Artificial Neural Networks (ICANN)*, Springer, 2013.

[318] S. Alghowinem, R. Goecke, M. Wagner, G. Parker, and M. Breakspear, "Eye movement analysis for depression detection", in *International Conference on Image Processing (ICIP)*, IEEE, 2013.

[319] E. Tafaj, G. Kasneci, W. Rosenstiel, and M. Bogdan, "Bayesian online clustering of eye movement data", in *Eye-Tracking Research & Applications (ETRA)*, IEEE, 2012.

[320] L. Yu and H. Liu, "Feature selection for high-dimensional data: A fast correlation-based filter solution", in *International conference on machine learning (ICML)*, ACM, 2003.

[321] C. Braunagel, E. Kasneci, W. Stolzmann, and W. Rosenstiel, "Driver-activity recognition in the context of conditionally autonomous driving", in *Intelligent Transportation Systems (TTS)*, IEEE, 2015.

[322] "Daimlers neuer hochdynamischer fahrsimulator in originalgröße - ein technischer überblick", *Acts INRETS*, 2010.

[323] M. E. Manaa and G. Abdulameer, "Web documents similarity using k-shingle tokens and minhash technique", *J. Eng. Appl. Sci*, 2018.

[324] C. E. Leiserson, R. L. Rivest, T. H. Cormen, and C. Stein, *Introduction to algorithms*. MIT press Cambridge, MA, 2001.

[325] A. Singhal *et al.*, "Modern information retrieval: A brief overview", *IEEE Data Eng. Bull.*, 2001.

[326] J. Otero-Millan, X. G. Troncoso, S. L. Macknik, I. Serrano-Pedraza, and S. Martinez-Conde, "Saccades and microsaccades during visual fixation, exploration, and search: Foundations for a common saccadic generator", *Journal of vision (JOV)*, 2008.

[327] M. B. McCamy, J. Otero-Millan, L. L. Di Stasi, S. L. Macknik, and S. Martinez-Conde, "Highly informative natural scene regions increase microsaccade production during visual scanning", *Journal of neuroscience*, 2014.

[328] A. Borji and L. Itti, "Defending yarbus: Eye movements reveal observers' task", *Journal of vision (JOV)*, 2014.

[329] F. M. Dekking, C. Kraaikamp, H. P. Lopuhaä, and L. E. Meester, *A Modern Introduction to Probability and Statistics: Understanding why and how*. Springer Science & Business Media, 2005.

[337] W. Fuhl, M. Tonsen, A. Bulling, and E. Kasneci, "Pupil detection for head-mounted eye tracking in the wild: An evaluation of the state of the art", in *Machine Vision and Applications (MVA)*, Springer, 2016.

[338] R. Muñoz-Salinas and R. Medina-Carnicer, "Ucoslam: Simultaneous localization and mapping by fusion of keypoints and squared planar markers", *Pattern Recognition*, 2020.

[339] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from simulated and unsupervised images through adversarial training", in *Computer vision and pattern recognition (CVPR)*, IEEE, 2017.

[340] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization", *Computing Research Repository*, 2014.

[341] M. Jiang, S. Huang, J. Duan, and Q. Zhao, "Salicon: Saliency in context", in *Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2015.

[342] J. Pan, E. Sayrol, X. Giro-i-Nieto, K. McGuinness, and N. E. O'Connor, "Shallow and deep convolutional networks for saliency prediction", in *Computer Vision and Pattern Recognition (CVPR)*, IEEE, 2016.

[343] W. Wang and J. Shen, "Deep visual attention prediction", *Transactions on Image Processing (TIP)*, 2017.

Online resources

[47] Tobii AB. (2020). Tobii pro vr integration, [Online]. Available: https://www.tobiipro.com/de/produkte/vr-integration/.

[48] Pupil Labs GmbH. (2020). Vr & ar - eye tracking - pupil labs, [Online]. Available: https://pupil-labs.com/products/vr-ar/.

[49] KM-ENKE GmbH. (2020). Vertigoggles, [Online]. Available: http://www.km-enke.de/en/vertigoggles-en/.

[50] Tobii AB. (2020). Tobii pro glasses 2, [Online]. Available: https://www.tobiipro.com/product-listing/tobii-pro-glasses-2/.

[248] J. Harel. (2020). A saliency implementation in matlab, [Online]. Available: http://klab.caltech.edu/~harel/share/gbvs.php.

[271] W. Commons. (2020). Tübingen - neckarfront 04, [Online]. Available: https://commons.wikimedia.org/wiki/File:T%5C%22ubingen_-_Neckarfront_04.jpg.

[293] P. Sohn and K. Reher. (2009). Partly cloudy, Walt Disney Pictures, [Online]. Available: https://www.pixar.com/partly-cloudy.

[332] K. Melnick, "2021 mercedes s-class will feature ar navigation and 3d eye-tracking", *VRScout.com*, Jul. 2020. [Online]. Available: https://vrscout.com/news/mercedes-s-class-ar-navigation-3d-eye-tracking/.

[333] S. Eye, "Smart eye is euro ncap ready", *Smart-Eye Blog*, Jun. 2020. [Online]. Available: https://smarteye.se/blog/smart-eye-is-euro-ncap-ready/.

[334] S. Hylèen, T. Goldbeck, and K. Messner, "Tobii paves the way for use of eye tracking in driver safety", *Tobii press release*, Nov. 2011. [Online]. Available: https://www.tobii.com/group/news-media/press-releases/tobii-paves-the-way-for-use-of-eye-tracking-in-driver-safety/.

[335] Bosch. (2020). Interior monitoring systems – how monitoring the vehicle interior increases safety, comfort, and convenience, [Online]. Available: https://www.bosch-mobility-solutions.com/en/products-and-services/passenger-cars-and-light-commercial-vehicles/interior-and-body-systems/interior-monitoring-systems/.

[336] S. Pinkow and M. Baum, "Alles im blick: Continental kombiniert umfeld- und innenraumkamera für automatisiertes fahren", *Continental press release*, Aug. 2019. [Online]. Available: https://www.bosch-mobility-solutions.com/en/products-and-services/passenger-cars-and-light-commercial-vehicles/interior-and-body-systems/interior-monitoring-systems/.

Appendix

Function	Parameter	Hyperparameter η	Initial η_0	Description
M_{left} M_{right}		$\eta_{t_{\text{fast2}}}$ left $\eta_{t_{\text{fast2}}}$ right	0 0	Threshold for FAST2 response on the left and right input image (4.1).
G_{left}	R_{left}	η_{R_x} left η_{R_y} left η_{R_w} left η_{R_h} left	$\frac{I_{\text{width left}}}{2}$ $\frac{I_{\text{height left}}}{2}$ $I_{\text{width left}}$ $I_{\text{height left}}$	ROI in which valid glint candidates are expected (4.2).
G_{right}	R_{left}	η_{R_x} right η_{R_y} right η_{R_w} right η_{R_h} right	$\frac{I_{\text{width right}}}{2}$ $\frac{I_{\text{height right}}}{2}$ $I_{\text{width right}}$ $I_{\text{height right}}$	
L_{BRIEF}	μ_{BRIEF}	η_μ mean BRIEF η_μ variance BRIEF	0 1	BRIEF similarity likelihood of a glint candidate from the left and the right stereo image (4.4).
	σ_{BRIEF}	η_σ mean BRIEF η_σ variance BRIEF	1 1	
L_{angle}	μ_{angle}	η_μ mean angle η_μ variance angle	$C_1 \lhd C_0$ $1\ ^{\circ 2}$	Likelihood of the angle between two matched 2D glint candidates (4.6).
	σ_{angle}	η_σ mean angle η_σ variance angle	$\propto$ RMS Error $1\ ^{\circ 2}$	
L_{dist}	μ_{dist}	η_μ mean dist η_μ variance angle	$\frac{v_{\text{max}}+v_{\text{min}}}{2}$ $1\ \text{px}^2$	Likelihood of the distance between two matched 2D glint candidates (4.10).
	σ_{dist}	η_σ mean dist η_σ variance dist	$\frac{\|\|v_{\text{max}}-v_{\text{min}}\|\|^2}{\log(256)}$ $1\ \text{px}^2$	
L_{IPD}	μ_{IPD}	η_μ mean IPD η_μ variance IPD	$63\ \text{mm}$ $1\ \text{mm}^2$	Likelihood of the distance between a 3D glint pair (4.12).
	σ_{IPD}	η_σ mean IPD η_σ variance IPD	$11.04\ \text{mm}$ $1\ \text{mm}^2$	
L_{yaw}	μ_{yaw}	η_μ mean yaw η_μ variance yaw	$0°$ $1\ ^{\circ 2}$	Likelihood of the vertical rotation of a 3D glint pair (4.13).
	σ_{yaw}	η_σ mean yaw η_σ variance yaw	$4.25°$ $1\ ^{\circ 2}$	
L_{roll}	μ_{roll}	η_μ mean roll η_μ variance roll	$0°$ $1\ ^{\circ 2}$	Likelihood of the lateral inclination of a 3D glint pair (4.13).
	σ_{roll}	η_σ mean roll η_σ variance roll	$4.25°$ $1\ ^{\circ 2}$	
L_{remote}	μ_{remote}	η_μ mean remote η_μ variance remote	$1000\ \text{mm}$ $1\ \text{mm}^2$	Likelihood of the distance from the 3D glint pair to the camera system (4.15).
	σ_{remote}	η_σ mean remote η_σ variance remote	$167.86\ \text{mm}$ $1\ \text{mm}^2$	

Tab. A.1: Overview of all adapted parameters η and the suggested initial value η_0.

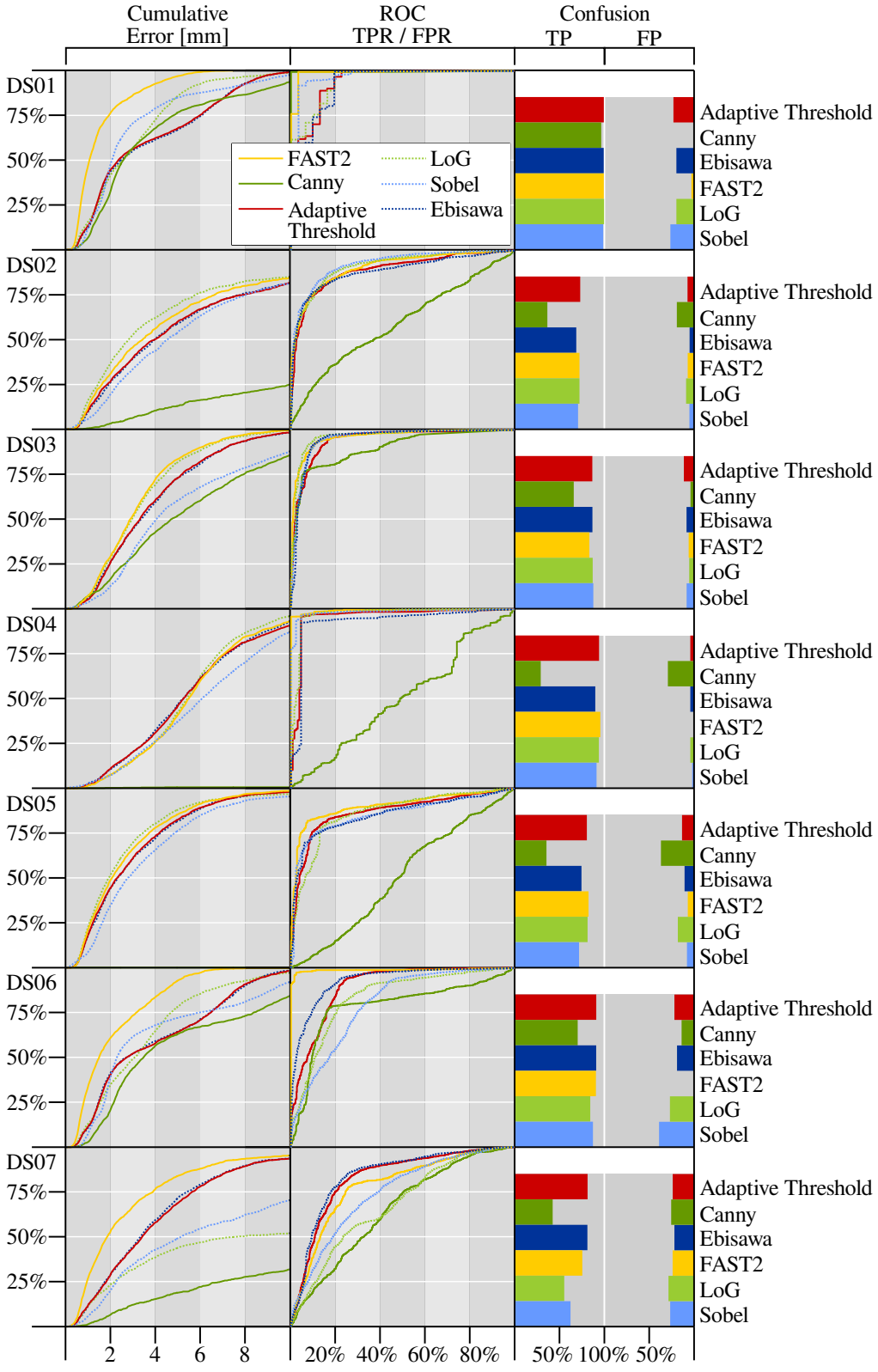


Fig. A.1: Evaluation of the glint filtering approach based on various feature extractors.

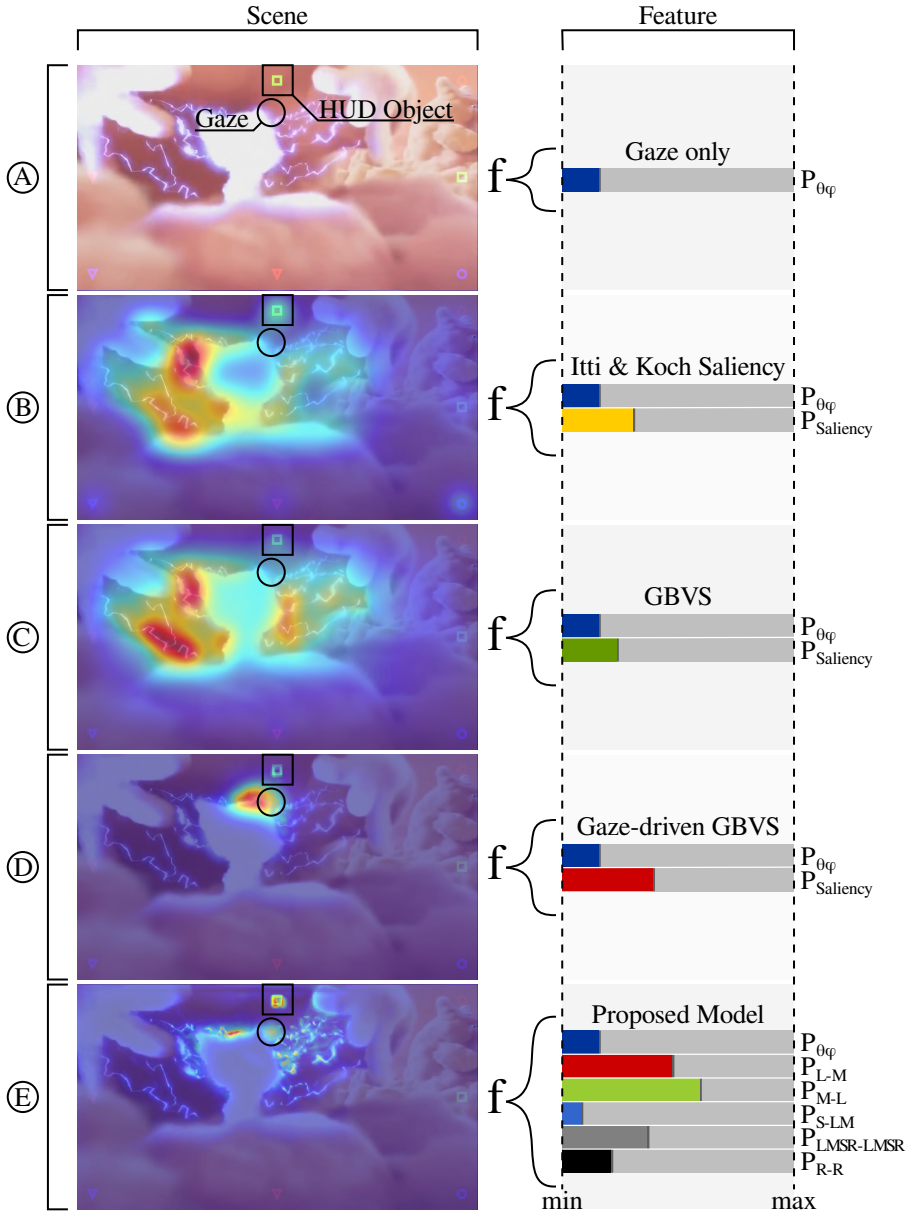


Fig. A.2: The first column shows an experiment scene: as a gaze only evaluation Ⓐ, and with heatmap overlays from the Itti & Koch saliency map Ⓑ, GBVS Ⓒ, gaze driven GBVS Ⓓ, and finally, the predicted Stimulus **I** (see Equation 6.37) of the proposed model Ⓔ. The second column shows the resulting feature vector **f** of a HUD object. For the gaze-only Ⓐ approach, only the eccentricity of the HUD object to the foveal axis is considered. Ⓑ-Ⓓ includes the output of the respective saliency map. Ⓒ shows the six-element feature vector of our model, as described in Equation 6.38

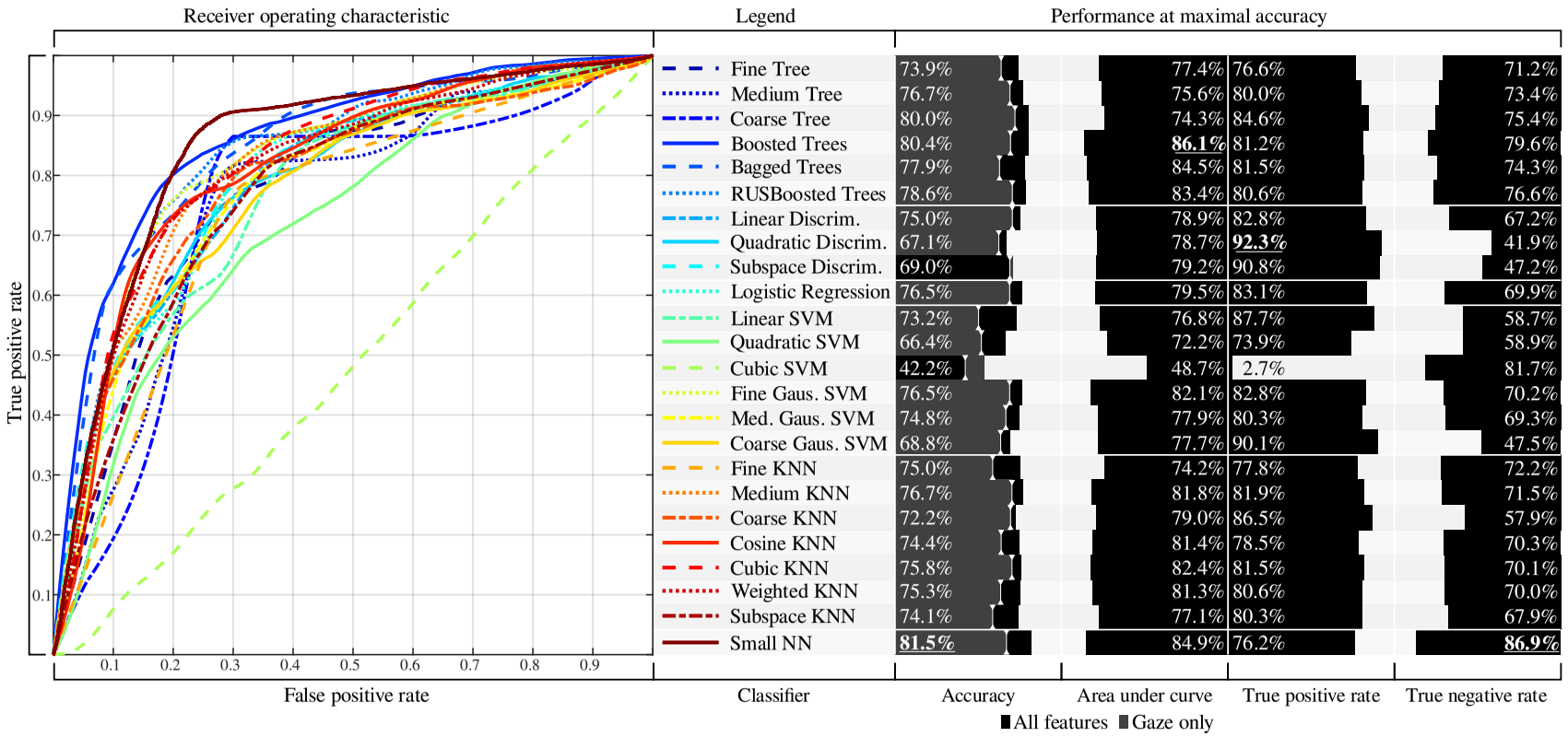


Fig. A.3: Classification performance of the six different classifier types in 24 different configurations. The left plot shows the different ROC curves for each classifier and their parameters indicated in the legend. Performances, shown on the right, are the accuracy of the features extracted by the proposed model (black), gaze only (grey), and using the saliency map from Itti & Koch (light grey). Except for the subspace discriminant classifier and the SVM with a cubic kernel, all classifiers show that they benefit from the additional stimulus information.

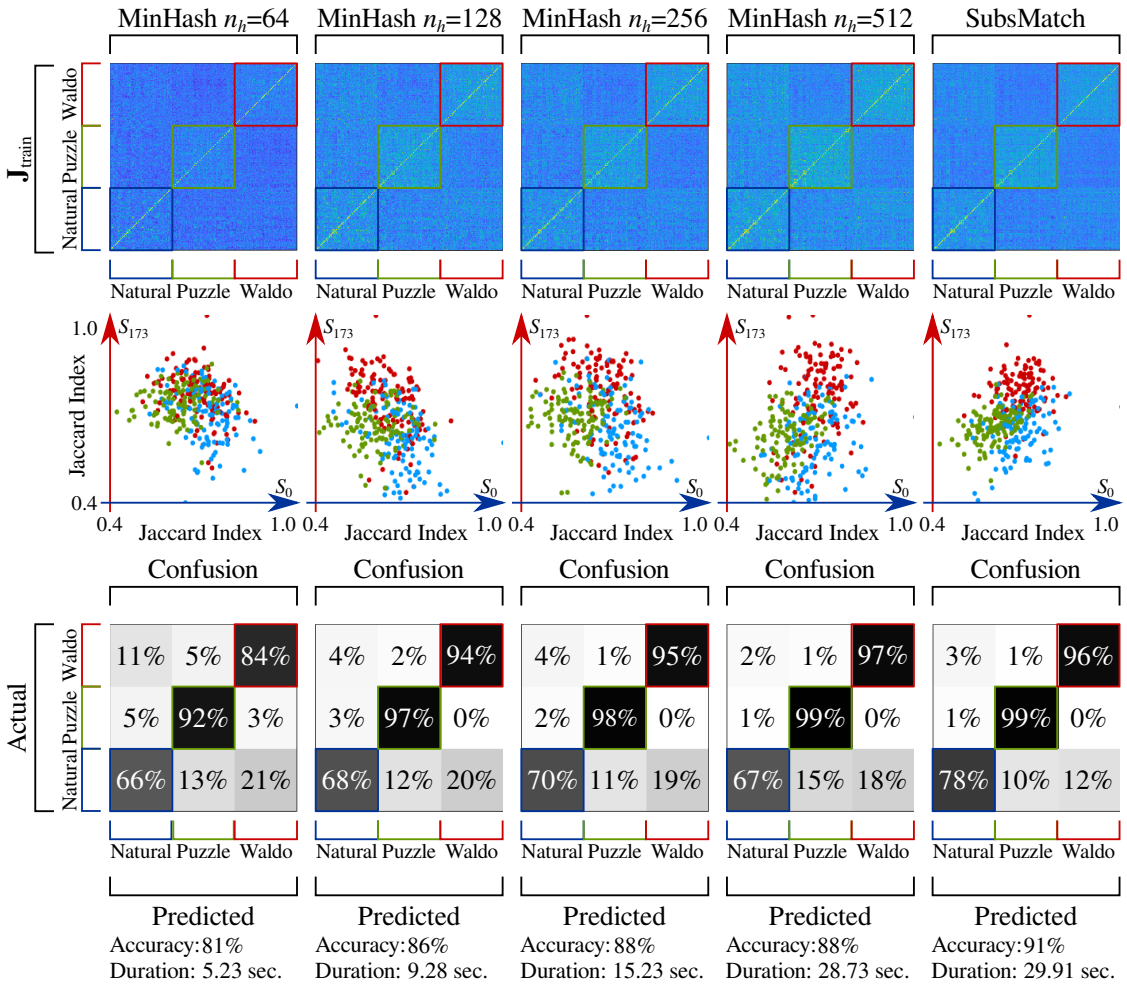


Fig. A.4: Each column represents a cross validation using the ETRA 2019 challenge dataset. The first 4 columns show the classification using MinHash across different numbers of hash functions. The last column shows the result of SubsMatch. The first column shows the pairwise calculated similarity of two scanpaths. Each row and column corresponds to one scanpath. For the classes Puzzle and Waldo, distinct clusters are recognizable. The second row shows the similarity values of the first row in the similarity domain of scanpath S_0 and S_{173}. Even when considering only 2 of 345 dimension, a clear cluster formation is recognizable with increasing number of hash functions. The last row shows classification performance of the cross validation as confusion matrix.

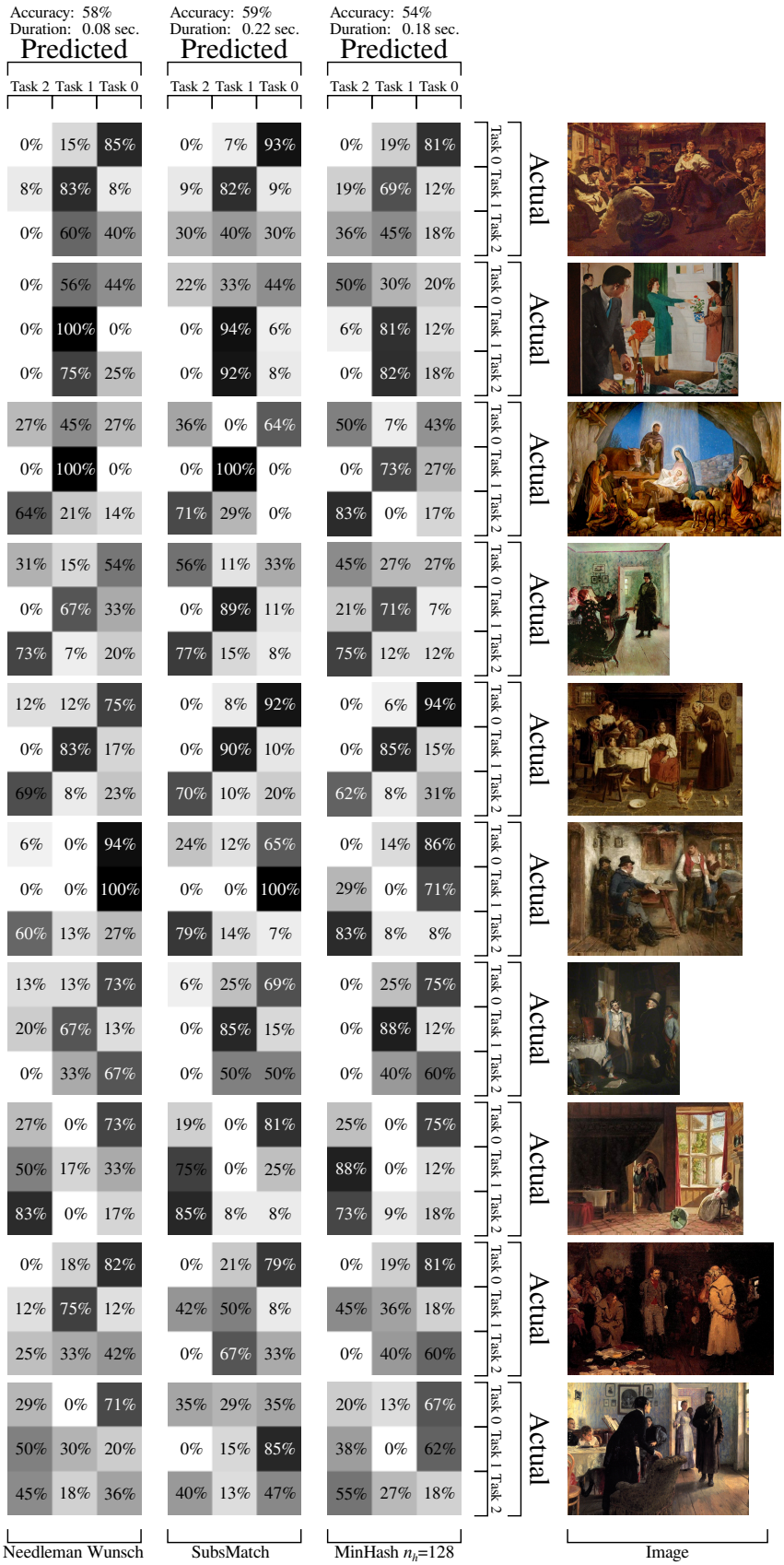


Fig. A.5: Evaluation of MinHash, SubsMatch, and Needleman-Wunsch. The first column shows the 10 presented stimuli [328]. For each stimulus, the subjects completed one of three tasks: (Task 1) Estimate ages of the people, (Task 2) Remember clothes, (Task 3) Estimate how long the visitor had been away. The following three columns show the confusion matrix for the classification of the task performed by the subject.