Institute for Computer Graphics and Vision
Graz University of Technology
Graz

Modeling Text Saliency in Human Visual Attention

Master's Thesis

Michael Schwarz, BSc.
m.schwarz@student.tugraz.at

February 2014

Supervision:
Univ. Prof. DI Dr. techn. Horst Bischof

Abstract

Information derived from human visual attention is important for many applications which need to focus on important regions in visual scenes. Currently used computational models of human visual attention provide possibilities to predict these most important regions for scenarios, in which eye tracker studies are not feasible. Even though researches already have identified the importance of text regarding the human visual attention, modeling text saliency is still a largely omitted topic. We provide a large scale database containing a combination of natural scene images and scene text images as well as according eye movement data collected at an eye tracking user study with 15 participants. A computational attention model based on a state-of-the-art approach to train specific context from eye movement data is learned on this new database to train an attention model with top-down text context. We further included a high-level text saliency feature to the computational attention model to test its influence on the prediction performance. The proposed model reports good performance results on our database as well as compared to other state-of-the-art models.

Keywords: Human Visual Attention, Computational Attention Model, Text Saliency, Text Detection, Saliency Model, Top-Down Text Context, Bottom-Up Text Feature.

Zusammenfassung

Für viele Anwendungen, die auf wichtige Bereiche einer Szene fokussieren, ist es essenziell, Informationen aus der visuellen Aufmerksamkeit des Menschen abzuleiten. Computergestützte Modelle der visuellen Aufmerksamkeit des Menschen sind in der Lage, diese Bereiche zu berechnen. Sie werden eingesetzt wenn Eye-Tracking Studien nicht sinnvoll durchführbar sind. Forscher haben den Einfluss von Text auf die visuelle Aufmerksamkeit des Menschen bereits entdeckt, aber die Modellierung von Textsalienz wurde bis jetzt wenig behandelt. Wir haben eine kombinierte Bilddatenbank aus generellen und textfokussierten Szenenbildern erstellt. Anschließend füllten wir sie mit Blickbewegungsdaten von 15 Teilnehmern einer von uns zu diesem Zweck durchgeführten Eye-Tracking Studie. Basierend auf einem Ansatz des aktuellen Standes der Technik, trainierten wir ein Text-Kontext fokussiertes computergestütztes visuelles Aufmerksamkeitsmodell. Wobei der Text-Kontext den Blickbewegungen unserer Eye-Tracking Studie auf textfokussierten Szenenbildern entspricht. In weiterer Folge fügten wir dem trainierten Modell ein neues Textsalienz-Feature bei, um dessen Einfluss auf die Leistung der Vorhersage zu überprüfen. Unser Modell zeigt gute Vorhersageleistungen sowohl im Bezug auf die von uns gesammelten Daten, als auch verglichen mit anderen aktuellen Aufmerksamkeitsmodellen.

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Graz, _____ _____

Date Signature

Eidesstattliche Erklärung[1]

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Graz, am _____ _____

Datum Unterschrift

[1] Beschluss der Curricula-Kommission für Bachelor-, Master- und Diplomstudien vom 10.11.2008; Genehmigung des Senates am 1.12.2008

Acknowledgements

First of all I would like to thank my supervisor Prof. Horst Bischof, who always offered me highly qualitative advice and showed a lot of patience for my occasional gaps of productivity. My special thanks also goes to Dr. Lucas Paletta, my advisor at Joanneum Research, who offered me the chance to work on this project, introduced me to the world of eye movements, visual attention and human factors, and supported me throughout the process and beyond.

This thesis was partially funded by the Institute for Information and Communication Technologies, JOANNEUM RESEARCH Forschungsgesellschaft mbH. It has also received funding from the European Union's 7th Framework Program under the grant No. 288587, and the Austrian Research Promotion Agency (FFG) under contract No. 832045, 'Research Studios Austria', FACTS, Human Factors Technologies and Services.

I would also like to thank my friends from the university as well as my colleagues from Joanneum Research, Martin Pszeida, Markus Blauensteiner, René Zmugg, and Gerald Fritz who listened to the problems I had, provided ideas, tips, practical help, or even distractions when I needed them. My gratitude also goes to the 15 participants of our eye tracking user study, who provided the most important data of this thesis. I am deeply grateful for all who proofread this thesis, most of Richard Bartle-Tubbs.

Furthermore I would like to thank my friends and my entire ultimate frisbee team for offering physical exercise, friendship and of course a lot of fun over all my years of study. My deepest gratitude goes to my parents and family for their emotional, organizational and intellectual support at all times.

And most importantly I would like to thank my girlfriend Pia for her support on all levels, her everlasting high spirits, and for her company on the several shorter or longer travels that temporarily distracted me from visual attention research.

Contents

Chapter 1

Introduction

A colorful poster on a blank wall, a lonely ship at sea, or a frisbee thrown in a group of people will likely catch your attention. But what exactly is attention? The first definition was introduced by William James in his work "The Principles of Psychology" in 1890 [James1890]:

> Every one knows what attention is. It is the taking possession by the mind, in clear and vivid form, of one out of what seem several simultaneously possible objects or trains of thoughts. Focalization, concentration, of consciousness are of its essence. It implies withdrawal from some things in order to deal effectively with others, and is a condition which has a real opposite in the confused, dazed, scatterbrained state which in French is called distraction, and Zerstreutheit in German.

James was the first to distinguish between intellectual attention that is directed at non-physically present ideal stimuli, and sensorial or visual attention, which focuses on present stimuli. Corbetta [Corbetta1998] further defined that:

> Attention defines the mental ability to select stimuli, responses, memories, or thoughts that are behaviorally relevant, among the many others that are behaviorally irrelevant. Selection is necessary because of computational limitations in the brainś capacity to process information and to ensure that behavior is controlled by relevant information.

Hence when we look at a scene our brain prioritizes objects, because the visual data is too immense to process all at once.

Since we need to see objects to trigger this sensorial attention, eye movements play an important role in visual attention research. Holmqvist et al. [Holmqvist2011] state that eye movements are controlled by muscles that enable the eye to move in a three dimensional orientation and that parts of the brain are controlling these muscles to locate the gaze at interesting stimuli. There are differently classified events in eye movements which can be measured with a device called an eye tracker. It tracks the motion of the eye in relation to the head or the gaze direction. According to Holmqvist et al. the most reported event is the so-called *fixation*, which is defined as the state whereupon the eye

rests on a position over a period of time, such as when we gaze at a word while reading for a certain time. Fast movements in-between fixations are defined as *saccades*, but even while we fixate on something the eye does not stay completely still and moves slightly performing so-called drift, tremor or microsaccade events. Holmqvist et al. further state that it is generally considered that visual attention is measured when a fixation occurs. The correlation between the processing of visual attention and eye movements is a topic that has been thoroughly researched since the late 1800s when the first eye trackers were build. According to Holmqvist et al., the earliest eye trackers were uniformly difficult to build and were in general invasive to the participants. Dodge and Cline [DodgeCline1901] proposed the method of photographing certain reflections from external light sources on the fovea, which has become the most used technique nowadays. In the mid 1970s, companies started to build and sell eye tracking devices to scientists and since then increasing numbers of manufacturers habe continuously improved these systems. Currently used eye tracking systems from companies such as SensoMotoric Instruments (SMI)[1], tobii[2] or SR Research[3], to name just a few, generally provide an algorithm that tracks the pupil and corneal reflection to estimate the gaze point [Holmqvist2011]. Pupil and corneal reflection tracking is used with a second reference point to compensate potential falsely detected head movements.

Based on Yarbus' [Yarbus1967] proposed correlation between attention and fixations, Just and Carpenter [JustCarpenter1980] introduced the hypothesis that comprehending a word takes a person as long as he or she fixates upon it. Hence they proposed that attention occurs when one fixates upon something. Research in visual attention modeling and eye tracking is commonly still based on this hypothesis, even though according to Hoffman [Hoffman1998] the current notion is that compared to the time the eye needs to move to the next fixation, attention is processed 100 - 250 ms faster.With his famous eye tracking experiments in which participants had to look at the painting "The Unexpected Visitor" by Repin with different tasks, Yarbus further showed the impact of a given task on eye movements. This led to the definition of the two main mechanisms of human visual attention: the *bottom-up* attention and the *top-down* attention. Bottom-up mechanisms are also called reflexive or stimuli-driven because they are triggered by the scene itself [Nothdurft2000]. They respond to objects in visual scenes humans find naturally or biologically interesting. On the other hand top-down mechanisms are also referred to as task-driven or voluntary because they are driven by cognitive factors [CorbettaShulman2002] such as tasks, expectations or knowledge.

The concept of selective attention models has became evermore interesting for other fields of research, inter alia computer science where systems often have to process large amounts of image data as fast as possible [Judd2011a]. Researchers in the fields of com-

[1]http://www.smivision.com/
[2]http://www.tobii.com/
[3]http://www.sr-research.com/

puter vision, robotics, computer graphics and usability, for example, have developed visual attention models in order to select the most important objects in visual scenes of any kind to improve their system's performance and computation time. For instance Visual attention models are used to select regions of interest or image segmentation in computer vision [Valenti2009] to measure the regional importance of meshes in computer graphics [Varshney2005], or for a movement decision system to select the most interesting locations in robotics [SiagianItti2009]. We will describe further applications using visual attention models in chapter 2.3.
These computational models of human visual attention were motivated by research findings of cognitive psychology described previously. In the field of human visual attention, *saliency* is commonly mentioned when addressing the most relevant parts of a scene. Therefore, we will stick to this naming for the rest of the thesis and describe the most important locations: "the most salient". Treisman and Gelade's feature integration theory [TreismanGelade1980], which introduced the idea of using different sensorial information as feature maps, was an important step for computational attention models and laid the foundation for many other models to come. Based on this theory, Koch and Ullman [KochUllman1985] proposed their algorithmic model of attention which provided efficient strategies to produce saliency maps with bottom-up cues. Itti et al. [Itti1998] provided a framework to compute so-called saliency maps that used the structure described by Koch and Ullman and which is still updated and used today. Saliency maps are defined as the resulting map of a computational attention model marking the most salient locations of a given image. They are usually presented as grayscale images where the image's intensity brightens correspondingly to the saliency of a pixel. The brighter the image, the more salient the pixel, as shown in figure 1.1. We give an overview on further computational

(a) Input image (b) Saliency map

Figure 1.1: A sample saliency map (b) for a given input image (a), the pixel's intensity relates to its saliency (brighter means more salient).

attention models based on bottom-up cues in chapter 2.1.1.

Eventually researchers added top-down cues to computational attention models which used high-level features, weighting values for certain features based on certain tasks or a combination of these. Though some bottom-up models achieved good results predicting human fixations, the general consensus is that a computational attention model combining bottom-up and top-down cues performs better [Henderson2005]. Judd et al. [Judd2009] used machine learning techniques to learn a model of attention using human fixations on a broad variety of scene images collected in an eye tracking study. Their approach provided a large database of human fixation data, as well as a combination of several well researched feature maps and a saliency classifier that is directly learned from human fixations. This state-of-the-art computational attention model outperforms most of the other currently used attention models, which is why we decided to use it as a basis for our approach in this thesis. Therefore, we will give a detailed description of the model in chapter 5. A more extensive overview of top-down attention models can be found in chapter 2.1.2.
High-level object features are already used in several attention models, however a focus on scene text is still missing. When an image containing text is shown to a person he or she will most likely read the text and therefore his or her attention will be drawn to these text areas. Cerf et al. [Cerf2009] did several experiments on how salient text, faces and cell phones are in human visual attention. They showed that faces and text have a strong impact on eye movements and added face and text locations as features to Itti et al.'s framework. Cerf et al. concluded that text features are an important aspect of visual attention and should be included in computational models to improve the prediction of human fixations. However, to our knowledge, using text features and the context that could be learned from machine learning models trained on text features are largely omitted aspects in current state-of-the-art computational attention models. Therefore, the aspect of text saliency should be further researched to improve currently available models and we will elaborate on this topic in chapter 2.2. Our hypothesis in this thesis is that including text features in a bottom-up way, as well as text context learned from these features as proposed by Judd et al., in short a focus on text saliency should improve the performance of predicting human fixations on scene images.

The condition of including text saliency in an attention model is first of all a text detection method to locate text areas in scene images. Text detection in natural images is an important task and a well-researched topic in computer vision. However, the variety of fonts and colors, lightning conditions or reflections, and the difference of indoor and outdoor environments pose several problems for a text localization, which are some reasons why the text detection problem is far from solved. Text detection algorithms can roughly be classified into region based algorithms that create connected components out of pixels with suitable parameters, and texture based algorithms, which classify pixel neighborhoods based on several text properties at several scales [Konuskan2008]. We summarize a short history and current state-of-the-art algorithms of this field of research in chapter 3.1. At the time this thesis started, the *Stroke Width Transform* (SWT) by Epshtein et al.[Epshtein2010] was a text detection algorithm that outperformed most of the other

state-of-the-art approaches. They introduced a new local operator (the SWT) to gather potential letter candidates from pixels, which their algorithm then filtered through several filter and heuristic stages to connect components and eliminate non-text regions. The SWT algorithm is able to detect scene text regardless of font, language, scale and whether the text is presented dark on bright or bright on dark. This is why at that time we decided to include Epshtein et al.'s approach as text feature in our computational attention model with focus on text saliency. A detailed description of this approach can be found in chapter 3.2. Since [Epshtein2010] was published, text detection approaches have, of course, advanced and several newer methods such as [Yin2013], [NeumannMatas2012], or [Yao2012] to name just a few, provide better detection performances at benchmark comparisons.
The methodology as to how our approach creates text saliency features is not dependent on a certain text detection algorithm; it could be easily changed. Our hypothesis for using different text detector algorithms is that the model's prediction performance should increase accordingly with an improved detection performance. Therefore, we also tried to train our computational attention model with an OpenCV[4] implementation based on the scene text algorithm proposed by [NeumannMatas2012] and the character candidates grouping by [GomezKaratzas2013], to show the impact a different text detector yields. These approaches are also further described in chapter 3. We compared different approaches of transforming the detected text areas of given images to text saliency maps: A binary saliency map where pixels inside a text bounding box are marked as salient, a text segmentation approach where segmented characters are marked as salient, and a novel approach to blur these text segmentation images to give a continuous saliency vote depending on the pixels distance to a character. Our novel approach yielded the best experimental results and will be described in chapter 5.1.2.

Judd et al. [Judd2009] collected an extensive database of general natural scene images and performed an eye tracking user study to gather human fixation data that they used to train their computational attention model. After a thorough examination of their image database we came to the conclusion that it contained very little scene text images. To train a computational attention model with more top-down text specific context we needed more human fixation data for scene text images. Therefore, we performed an eye tracking user study ourselves on an extensive text database collected from images of the well known ICDAR robust reading competition[5]. However, since our model should not only perform well on text images, we combined the database with Judd et al.'s original database so we would not lose their general natural scene image context. The user study, the combined database and the way we transformed the eye tracking data to usable human fixation maps is described in chapter 4.

The contributions of this thesis are the following:

[4]http://opencv.willowgarage.com/
[5]http://algoval.essex.ac.uk/icdar/Competitions.html

We performed an **eye tracking user study** to gather text specific as well as general context for natural scene images.

For this study we created a combined image database with images from a scene text image database and a general scene image database to acquire **human fixation maps** from both of them.

We included a **text saliency feature** to a state-of-the-art computational attention model, using a text detection algorithm and a novel approach to transform detected text regions to saliency maps.

We learned a general **computational attention model** with newly collected top-down text saliency context based on the framework of a state-of-the-art computational attention model.

As described in chapter 6, the results of our experiment have shown that including text saliency to Judd et al.'s computational attention model increased the model's performance. We outperformed their original model comparing the model's true positive rate when matching predicted saliency maps against actual human fixations at different percentages of saliency, as it is common for applications that use computational attention models. We also achieved an improvement comparing the Receiver Operating Characteristics (ROC), however, one not as significant as we expected, when the original Judd et al. model was retrained on our dataset. We concluded that this is mainly because of the fact that their approach was retrained on our database containing enough text context and the fact that text saliency feature alone did not change the performance as significantly as expected. Also the center bias that occurs when people take photos of natural scenes and important objects tend to be oriented in the middle of the image, and the features used by Judd et al. that already emphasized text properties, lowered the overall performance difference of the proposed model and theirs.
To compare our new models' performance to other models as well, we also carried out a benchmark comparison on the saliency benchmark[6] by Judd et al. [Judd2012], which tests three different metrics, and scored the fourth best rank of 22 models. Overall, we provide a robust new computational attention model based on the approach of Judd et al. [Judd2009], which provides an efficient way to compute saliency maps for natural scene images focusing on text saliency.

[6] http://people.csail.mit.edu/tjudd/SaliencyBenchmark/

Chapter 2

Computational models of human visual attention

In this chapter we are giving a background to the existing computational models of human visual attention. We will start by exploring the differences between common bottom-up and task based top-down models in section 2.1. In this section we will further explain why we have based our project on the work of Judd et al. [Judd2009]. Since the focus of this thesis is text saliency, we will then explain why text saliency is an important part of visual attention in section 2.2. After these sections, we will complete this chapter with a description of the applications of computational visual attention models in section 2.3.

2.1 Bottom-Up or Top-Down models

There is an increasing interest in systems that model human visual attention in the fields of computer science. These systems are commonly used as mechanisms to select the most relevant objects in a scene containing visual data. In computer vision, human computer interaction and robotics for example, computer systems sometimes have to focus on relevant objects. A computational model of visual attention can help to decrease the computing time of these systems by focusing on relevant objects. Therefore, the scope of these systems for computer scientists is predominantly to improve vision based systems by obtaining a numerical likelihood model of the most relevant parts in a scene. All of these systems, though, are based on psychological theories of human visual attention. Therefore, in order to implement a human visual attention model, we have to try to better understand the human perception.
Generally, there are two types of approaches for computational visual attention models: bottom-up and top-down models. The difference between these two is based on the mechanisms of visual attention. Bottom-up models predict salient regions based on the visual scene only[Nothdurft2000]. In this case the mechanism is called stimulus-driven or automatic. In comparison, top-down models add the influence of a task and are driven by cognitive factors. These factors could be goals, knowledge or expectation of a task [CorbettaShulman2002] and the mechanism is called voluntary or goal-driven.

2.1.1 Structure of Bottom-Up attention models

Figure 2.1 shows a flow diagram of a typical feature-based bottom-up attention model, by Itti and Koch [IttiKoch2001]. As stated in Judd et al. [Judd2011a], this attention model structure is adapted from Treisman and Gelade's feature integration theory [TreismanGelade1980], and the guided search model by Wolfe et al. [Wolfe1989].

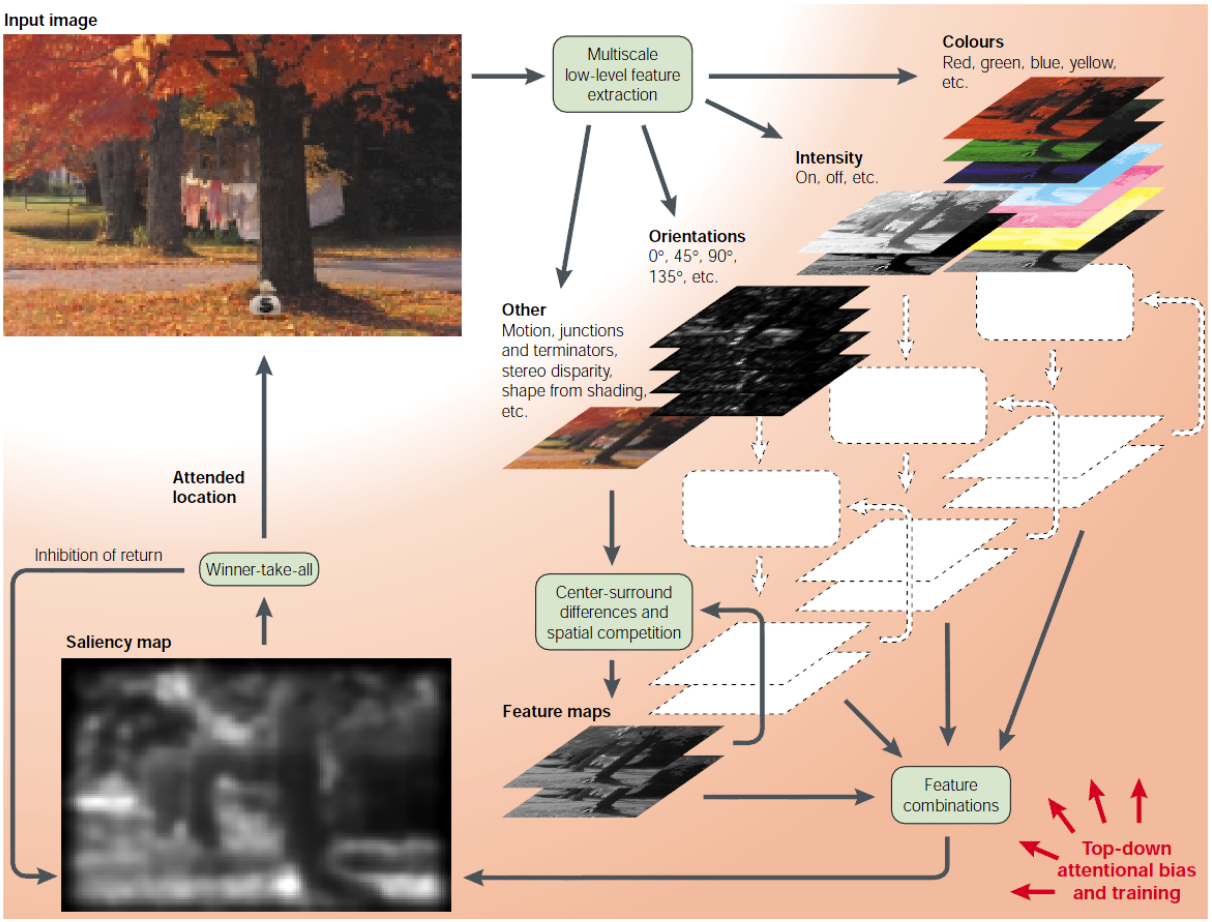


Figure 2.1: A Flow diagram of a typical bottom-up attention model. From Itti and Koch [IttiKoch2001].

As we can see in this flow graph, the basic idea is that the combination of several pre-attentive feature detection mechanisms lead to an overall saliency map. This flow graph is based on Koch and Ullman's algorithmic model of attention [KochUllman1985]. Their hypothesis is that a centralized saliency map can provide efficient strategies for the deployment of attention, on the basis of bottom-up cues.
According to Judd et al. [Judd2011a] a common structure for one input image, based on this hypothesis, would be the following:

Compute image pyramids to enable feature computation for different scales.

Compute image features such as color, orientation and intensity. Usually the image features are subdivided into more sub-channels, such as red, green and blue color-maps for the color feature.

Apply differences of Gaussian or center surround algorithms to get the inter-map contrasts into the feature maps. These algorithms compare the mean values of surrounding regions to the means of center regions and create sub-feature maps.

Create conspicuity maps which are the result of summing up these sub-feature maps.

Normalize the conspicuity maps to make their values comparable to other conspicuity maps. If they were not normalized, then channels with more features would be ranked higher.

Weight the conspicuity maps to define their feature importance. This is normally done by using a weighting function for each conspicuity map.

Combine the maps to get the image's **saliency map**. Usually a linear summation of all feature maps is carried out. A gray-scale image is the result, wherein the most brightest pixels are the most salient.

Koch and Ullman [KochUllman1985] introduced a structure of a bottom-up computational attention model, based on Treisman and Gelade's feature integration theory [TreismanGelade1980]. It was not implemented back then, but it provided the basic algorithms for implementations. One of the first implementations of this model was done by [ClarkFerrier1989]. It provided the above mentioned feature maps, a weighting system and a resulting summed up saliency map. An important work based on this model is the *iLab Neuromorphic Vision C++ Toolkit*[1] (iNVT) by Itti et al. [Itti1998]. This toolkit was updated several times [IttiKoch2001], [NavalpakkamItti2006] and is still kept up to date by Itti et al.
An improved and more stable version of this toolkit was introduced by [DraperLionelle2005], called SAFE (selective attention as a front end). They found errors in the fields of geometric transformations in the iNVT toolkit, and released a refactored version. Some researchers extended or altered the toolkit with different approaches to improve the performance of predicting human fixations. For example [WaltherKoch2006] introduced the Saliency-ToolBox (STB)[2], by adding so-called proto-object features, which are objects of volatile units of visual information. A more recent extension was done by [Valenti2009], who added isocentric curvedness and color features to the iNVT model. This approach was based on the idea that attentive objects often have local characteristics which differ from the rest of the image. Furthermore, [Vasconcelos2010] introduced a spatiotemporal saliency model, based on their center-surround framework [Vasconcelos2009]. Their approach extends a center-surround saliency model, which has been proposed for static images, to an unsupervised spatiotemporal saliency algorithm, usable for scenes with dynamic backgrounds. Although it is known that task specific top-down cues have a great influence on our vi-

[1] http://ilab.usc.edu/toolkit/
[2] http://www.saliencytoolbox.net/

sual attention [Yarbus1967], most models only cover the aforementioned bottom-up cues. Bottom-up models are of course much easier to model, as top-down modeling requires task specific knowledge.

2.1.2 Top-Down components of attention models

Figure 2.2, the "Unexpected Visitor" by Yarbus et al. [Yarbus1967], shows how eye movements change when viewers have different tasks.

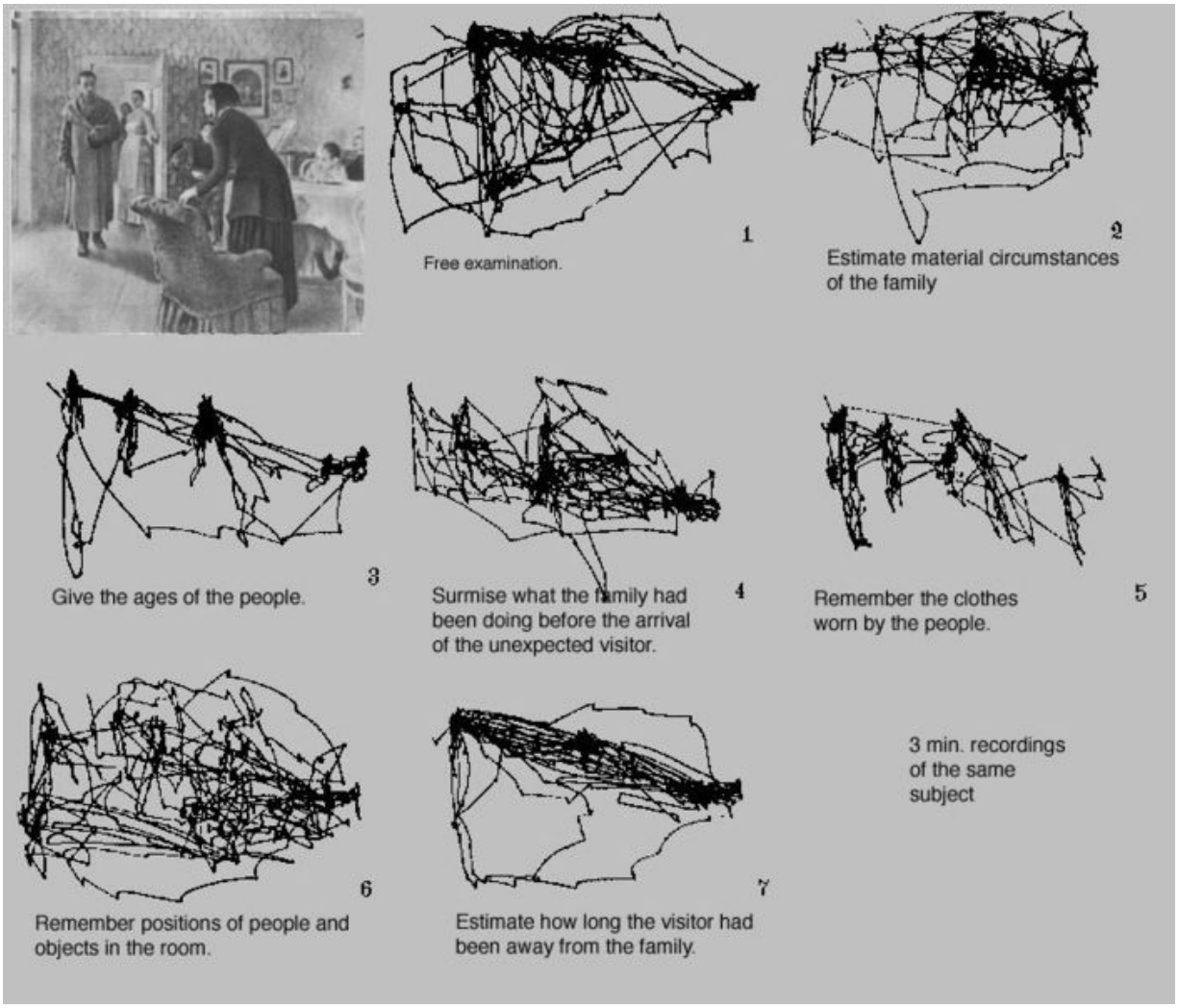


Figure 2.2: The famous "Unexpected Visitor" by Yarbus et al. [Yarbus1967], first showed that eye movements change with different tasks.

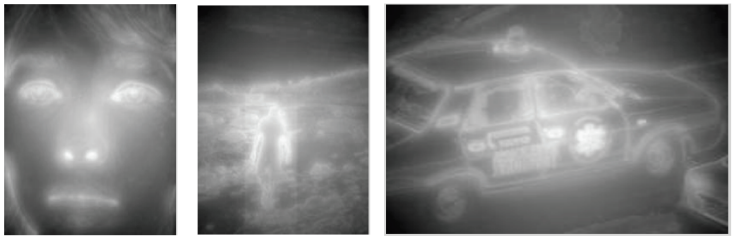
The idea of including more knowledge than only bottom-up cues was introduced by the guided search model and Desimone and Duncan's biased competition theory [DesimoneDuncan1995].

Henders et al. [Henderson2005], have shown that the influence of low-level information for search tasks is minimal, and that top-down information dominates a search task. Therefore, a good attention model should include top-down components when tasks are given. Torralba et al. [Torralba2006] have shown further that context in real-world scenes is very important for attention modeling. Their approach is a combination of bottom-up saliency and scene context for guiding search. When predicting human fixations for a search task, it outperforms bottom-up saliency-based models. They provide a weighing system that weights high salient areas within a selected region higher than regions outside of the selection. This approach was updated by [Hidalgo2009] and is used, for example, in the approach of Judd et al. [Judd2009] and in our approach in this thesis. A more recent approach which used context-aware saliency was done by [Goferman2010]. This approach detects the representing image regions, not only salient objects. They include global effects on frequently occurring objects and add a face detector.
Another structural possibility for adding top-down components is modifying a feature map's weight, depending on the given task. This concept was introduced by [Wolfe1989], a model that increases the weight of features which are important to a given task. Their approach is used in several more recent approaches such as [ElazaryItti2010] and [IttiNavalpakkam2010]. When using a database containing many annotated images, [Marchesotti2009] proposed an approach based on the concept that images with similar appearances are more likely to have similar saliency. A classifier can create the saliency maps, when the annotated image database is provided. Yet another way to add top-down components is to include object detectors, when the given task focuses on certain objects. The approach of [Cerf2009] showed that faces and text are very attentive features for humans, which can not be easily ignored. To improve the model of [KochUllman1985], Cerf et al. included a face detection conspicuity map. Their face detection was done based on the approach of [ViolaJones2002] and these new features significantly improved the performance of predicting human fixations.
A more recent general model, which used both bottom-up and top-down cues, is introduced by Judd et al. [Judd2009], [Judd2011a]. They have provided a supervised learning model of human visual attention, which combines image-based information (bottom-up) with semantic-based cues (top-down). The proposed model uses a lot of the previous stated approaches such as biologically inspired features [KochUllman1985], [WaltherKoch2006], object-based [Cerf2009], context-based features [Torralba2006], amongst others. All in all they combined 33 features (including sub-channels): sub band features, color features, Itti and Koch channels (using the STB by [WaltherKoch2006]), distance to the center, automatic horizon, face, person and car detectors. They used a support vector machine to train the model, using positive and negative samples of each feature-channel. Judd et al. [Judd2011a] has stated several times that center bias is an important aspect of computational attention models. Center bias means that human fixations are usually biased towards the center of an image [ParkhurstNiebur2003], [BruceTsotsos2009], [VincentTatler2009], [Tseng2009]. Even though this bias was researched often, [Judd2009], [ParkhurstNiebur2003] and [ZhaoKoch2011] are the only ones who have implemented models that included this bias. To our knowledge the approach of Judd et al. [Judd2009] still

outperforms the currently used and available computational attention models for general scenes. This is why we have use the framework of [Judd2009], as basis for the computational model of this thesis. Figure 2.3 shows a sample of resulting predictions using the model of Judd et al. [Judd2009].

(a) Scene images

(b) Human fixations



(c) Model predictions

Figure 2.3: Images by Judd et al. [Judd2009]. Samples of predictions where people look (c) as saliency maps, provided by the model of Judd et al., compared to real human fixations (b).

2.2 Importance of text saliency in visual attention

Image features are important in order to identify salient locations in a scene, as described in section 2.1.2. Computational models of attention are often provided with different image features to improve their performance in predicting human fixations. As stated before, many models have already included image features such as object-based or context-based features (for example [Judd2009]). Cerf et al. [Cerf2009] have shown that text is a very

attentive feature for humans, as well as faces. They carried out a user study with four different experiments, testing how attentive faces, text and cell phones are to humans. Their psychophysical results show that text and faces have an impact on eye movements, which are rapidly attracted to both. Furthermore, they refined the iNVT model of [Itti1998], and added high-level semantic information such as text and face locations. With this refinement they enhanced the model's performance in predicting human fixations. Hence, they concluded that scene text is an attentive feature, which should be used to improve human fixation predictions. However, scene text features, to our knowledge, are still largely omitted in computational attention models, especially in newer models such as the current state-of-the-art model by [Judd2009]. More recently, [Shahab2012] evaluated four computational attention models ([Itti1998], [Harel2007], [Torralba2006], [Zhang2008]), on their behavior at text regions. Their conclusion was that the saliency maps provided by these models could be used to improve scene text detection methods by suppressing non text regions. Judd et al. [Judd2009] also stated that people commonly look at text, faces or other persons but did not include a text feature to their computational attention model. Since, to our knowledge, text saliency has not yet been included in state-of-the-art computational model of attention, we propose that including scene text features, provided by a state-of-the-art text detection algorithm, would further improve the performance of the model.

2.3 Computational visual attention applications

Whenever we need to extract the most interesting data from an image scene, a computational visual attention application is very useful. Several fields in computer sciences, such as computer vision, computer graphics, robotics or human computer interaction, use these applications. Of course, for the means of usability testing or marketing research, visual attention aspects are also very important. As [Judd2011a] states, an attention selection provides an intuitive way for selecting regions of interest in image scenes.
In the fields of computer vision, salient regions provide natural regions for image segmentation or region of interest (ROI) calculation. Both [Valenti2009] and [Achanta2008], for example, proposed an image segmentation approach based on a saliency model. Some researchers use bottom-up saliency information for improving object recognition methods. The idea is to combine a saliency based model with an object recognition system, as in the approaches of [Miau2001], or more recently [WaltherKoch2006]. Mahadevan et al. [Mahadevan2013] proposed an approach to biologically inspired object tracking, using center-surround saliency mechanisms [Vasconcelos2010]. They used a saliency model for the learning stage, by combining a bottom-up saliency and a focus of attention method, to find interesting features for target detection. In the detection stage they used a top-down saliency detector, focused on the target, combined with a feature based saliency method. Their experimental results show that their method outperforms several state-of-the-art object trackers.

In computer graphics computational models, for example, can again be used to identify ROIs in a scene. Santella et al. [SantellaDeCarlo2002], have shown that ROIs could help detecting the appropriate level of detail needed for non photo realistic renderings, such as abstract photographs or stylization. Saliency models can also be used for measuring the regional importance of meshes. For example, [Varshney2005] used low-level human visual attention cues to compute a mesh saliency operator. Another application is to provide estimations for computing the best crop of a scene, as in the approach of [Santella2006]. They provided an interactive method for image cropping, based on fixation data.
In robotics a very important task is for the robot to decide where to look next. Clark and Ferrier [Clark1988] introduced the first so-called active vision system with visual attention. This system described how to direct a robotic head to potential interesting objects in a scene. Their approach performed a simple tracking after fixating the most salient region in an artificial scene. More recent systems pertaining to directing attention are done by [Butko2008] and [Zhang2008]. Usually robots use a laser scanner approach for their localization. However, this approach sometimes fails in outdoor environments, and a visual approach, using landmarks, is applied. To support these systems in the detection of landmarks, computational saliency models, which select ROIs in sensor data, can be used. The proposed landmarks can be salient objects in the environment, such as in the approach of [SiagianItti2009].
Eye tracking devices are often used by companies in the fields of marketing and usability, as can be seen in [PooleBall2006]. Companies are mostly interested in at what and where people look on their web pages, and how to accordingly place their products. Therefore, they usually want saliency maps for their individual setup. SMI[3] or tobii[4] provide software solutions for creating such saliency maps derived from human fixations, gathered by user studies. However, to our knowledge, there are no general computational saliency models yet in the fields of usability or marketing that could provide saliency maps without large user studies.
An important aspect of the reasons for our use of computational models of attention, is that they usually decrease the computation time of the approach we combine it with. This is possible due to the underlying nature of a computational model of attention: in predicting or focusing on the most salient regions. All these approaches above show that there is a variety of fields where computational models can be applied. This thesis is about the potentials for improvement in general computational model of attention through the application of good text features. Hence, we will further discuss text detection algorithms in chapter 3.

[3]http://www.eyetracking-glasses.com/products/eye-tracking-glasses/technology/
[4]http://www.tobii.com/en/eye-tracking-research/global/research/usability/

Chapter 3

Text detection

Text detection in natural scene images is an important task in computer vision. Many tasks such as robotic navigation or aiding software for the visually impaired, for example, need a robust text detection method for natural scenes. Detection texts in different fonts, colors, environments (indoors and outdoors), however, lay out several complex parameters for good results. We already discussed the benefits of scene text features for computational attention models in section 2.2. Furthermore, we already stated that we included a text detection feature in our computational attention model. Therefore, we will now examine the topic of text detection further. In this chapter we will first give an overview to the currently used text detection approaches in section 3.1. Since we have used Epshtein et al.'s [Epshtein2010] approach in this thesis, we will describe it in-depth in section 3.2. We will also give a performance comparison between state-of-the-art text detection algorithms and show why we have chosen Epshtein et al.'s approach for our thesis. However for the method of computing our text saliency features it does not matter which text detector algorithm is chosen as long as it provides rectangular bounding boxes for found text locations. The hypothesis of course is that a better performing text detection algorithm will also achieve a better performing computational text saliency model. Therefore, we also included a second text detector that will also be described in this chapter, to show the impact on the computational attention model when using a different text detector.

3.1 Overview of algorithms

Text detection algorithms can be roughly grouped into the following: region based algorithms and texture based algorithms. We adapted the following differentiations between those methods, from [Epshtein2010] and [Konuskan2008].
Region based algorithms group pixels with properties such as constant color together. These groups create connected components that are then filtered to eliminate non-letter components. The analysis of which components might fit together can be done using edge detectors or by analyzing the aspect ratio of edges for example. The advantages of this approach are that the detection can not only handle horizontal texts, but different angles, and the detection is not dependent on scale.

Texture based algorithms, on the other hand, classify the pixels neighborhoods on several scales. The classification is based on several text properties such as the variance of intensity, gradient changes, or edge density. Hence, these approaches build on the presumption that text regions have distinct properties in texture, which discriminate them from their background. These approaches are usually computationally more complex than region based methods because the image has to be scanned for several scales. Furthermore, typically only scaled down or small texts contain the required texture properties, which might affect the detection precision. And finally texture based methods usually have difficulties detecting angular text.

We will now give an overview of approaches published in both categories, starting with region based methods. Li et al. [Li2000] proposed a scale space feature extractor that uses an artificial neural processor for text detection in videos. Their approach decomposes an input image into smaller parts, and uses a feature based approach to group potential connected text regions. Ye et al. [Ye2005] introduced a coarse-to-fine method to detect text lines with complex backgrounds. They used multi-scale wavelet features for a coarse detection, which located potential text pixels, followed by a density based region growing algorithm to create connected components. A similar approach was proposed by [Gllavata2004], who used high-frequency wavelet coefficients to characterize text and background regions. Both approaches use wavelet features to distinguish between text regions and non-text regions, however, they apply different classifiers to actually detect scene text in images. As already stated we are using Epshtein et al.'s Stroke Width Transform approach [Epshtein2010] in this thesis. Their approach falls into the category of region based methods, however, they used a pixel wise stroke width measure instead of the common edge, intensity or color similarities. Essentially, they introduced a local operator (the stroke width transform) to gather potential letter candidates from pixels, followed by several filter and heuristic stages to connect components and eliminate non-text regions. The proposed algorithm is able to detect scene text regardless of font, language, scale and whether text is presented dark on bright or bright on dark. At the time we started this thesis the Stroke Width Transform approach was the most promising text detection algorithm, which is why we decided to use it for the text saliency implementation. An in-depth description of the algorithm can be found in section 3.2 and a further description on our text saliency implementation will follow in chapter 5.1.2. A related approach to Epshtein et al.'s is the Character-Stroke Detection for Text-Localization and Extraction [Subramanian2007]. They use the knowledge that text has a roughly constant stroke width and proposed a fast and robust method to detect character stroke widths. Epshtein et al., however, states that there are several differences between these two approaches. First of all, Subramanian et al.'s method scans the image horizontally, while looking for intensity changes and always assuming dark text on bright backgrounds. Found regions are then analysed for stroke width and color constancy, also a certain range of stroke widths is not learned but assumed to be known.

The maximally stable extremal regions (MSER) approach by Matas et al. [Matas2004] was used as the basis for several novel text location and text segmentation approaches. Originally the MSER algorithm was introduced as a blob detector to establish correspondences between stereo-images. The new set of image elements called extremal regions (ERs) had several properties, such as the affine invariants or lighting sensitivity (monotonic transformation of the image's intensity), which were also useful for text detection approaches. Neumann and Matas [NeumannMatas2011] introduced a general method for text detection that departed from a strictly feed-forward pipeline and proposed a method to simultaneously process multiple text lines. They used synthetic fonts to train the algorithm and exploited the MSER approach to collect data that is robust to geometry and lighting. The properties of the ERs are used to perform a robust character recognition. They further proposed a scene text detection approach [NeumannMatas2012] using a real-time character recognition based on ERs. The OpenCV[1] library implemented a scene text detection algorithm based on this approach and combined it with the multi-script text extraction from natural scenes approach by Gomez and Karatzas [GomezKaratzas2013]. We used this implementation to compare a more recent approach to the results we achieved using Epshtein et al.'s Stroke Width Transform approach and will therefore describe it in more detail in section 3.3.

Texture based methods commonly use texture analyzing methods such as wavelet decompositions [Gllavata2004], Gabor filtering [JainBhattacharjee1992], [Mancas-Thillou2007] or gaussian filtering [Wu1997], for example, to filter textural information. Usually these filters are applied to the input image to obtain texture features and then compute the contrast, correlation, entropy or energy, which are then stored as a feature vector. The feature vectors are then used to classify text from non-text regions with the aid of machine learning algorithms. Jain and Bhattacharjee [JainBhattacharjee1992], for example, proposed a multi channel Gabor filter to gather texture information and computed the energy around filtered pixels. To classify the text regions they used a squared error cluster method on the computed features. A more recent approach by Thillou et al. [Thillou2005] used the same texture segmentation as [JainBhattacharjee1992], but changed the machine learning algorithm to an unsupervised k-means approach. Chen and Yuille [ChenYuille2004], proposed a well performing approach to detect scene text on cluttered backgrounds. They used an AdaBoost learning algorithm on prior determined potential text regions. A success rate of 93% was reported for 35 images, however, no comparison with other approaches on a database was given.

Some researchers combine the efficiency of region based methods with robust texture based ones. Liu et al. [Liu2005] proposed a text detector for color images with complex backgrounds, using a connected component analysis to provide text candidates, followed by a candidate verification with texture features. More recently a well performing approach was introduced by Lee et al. [Lee2011], who also used a combination of texture based

[1] http://opencv.org/

methods and connected components. They proposed a novel approach which extracts six different text feature classes and used an AdaBoost algorithm with a multi scale sequential search. Their feature classes included: variance and expectation of x-y derivatives and the local energy of the Gabor filter as proposed by [ChenYuille2004]. These are combined with a statistical texture measure of image histograms, a simple edge detection, a computation of variance of wavelet coefficient, and finally a connected component analysis. A resulting feature map is then scanned with a 64 x 48 search window on a multi scale sequential level. Each of these windows are trained with an effective AdaBoost classifier that computes whether it is a text window or a non-text window. A comparison of performance on the ICDAR [Lucas2003] database, showed that Lee et al.'s approach outperforms several state-of-the-art text detection methods, with a precision of 66%, a recall of 75% and the combined quality measure of 70%. As the reader will notice later in section 3.2, this is even a better score than the stroke width transform algorithm of Epshtein et al. reaches. Since this approach was published after we started the project for this thesis, we had no knowledge of a better approach than that of Epshtein et al. Therefore, we decided back then to use this state-of-the-art algorithm as a text detection feature for our computational attention model.

Several other approaches with state-of-the-art localization and detection results have been published in recent years, such as [Yin2013], [NeumannMatas2012], [YiTian2011], or [Yao2012]. More comprehensive studies on different text detection algorithms are provided in surveys such as [Jung2004] and [Liang2005]. In the following chapter we give a detailed description of the stroke width transform algorithm, including a performance comparison table.

3.2 Stroke width transform

The text detection algorithm we have used in this thesis is the stroke width transform (**SWT**) method by Epshtein et al. [Epshtein2010]. The SWT algorithm contains the following steps in its workflow:

- Computing an **edge map** of the input image
- Computing the **SWT** operator
- Finding suitable **letter candidates**
- **Filtering** these candidates
- **Aggregating** text lines out of the filtered letter candidates
- Using heuristics to break text lines into **detected words**

We will now explain these steps in more detail.

The SWT algorithm:
SWT is a local image operator and calculates the most likely stroke a pixel is part of. A stroke is defined as a contiguous part of the image that has an almost uniform width. Figure 3.1, taken from [Epshtein2010], shows an example of the SWT implementation. The

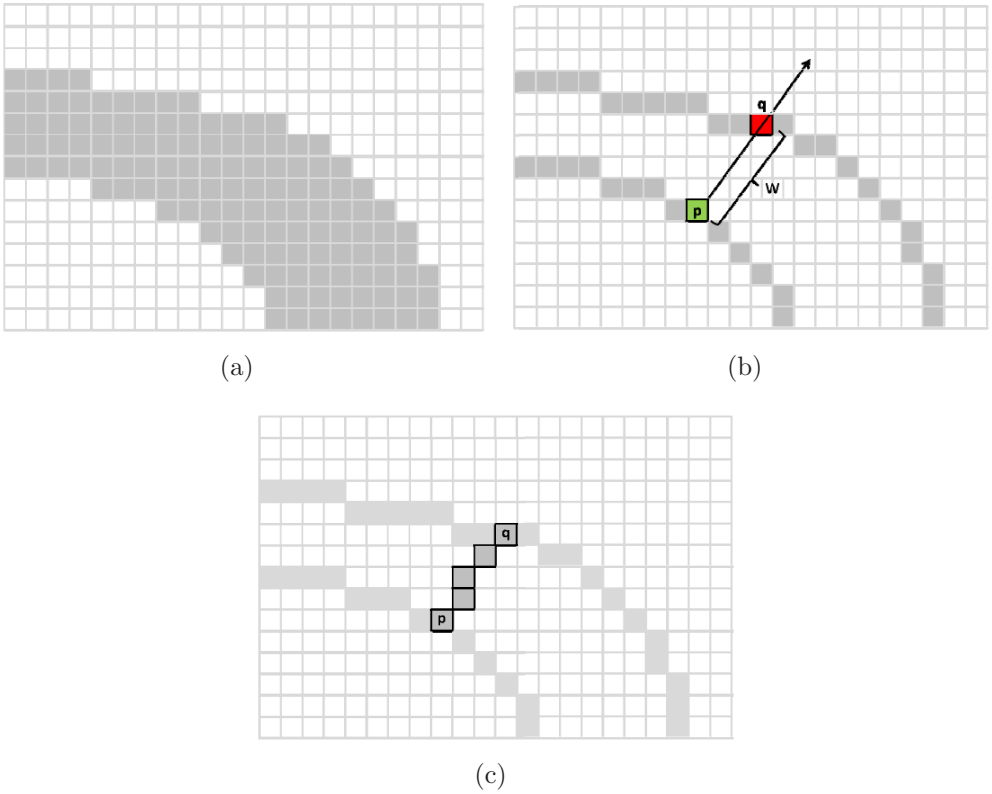
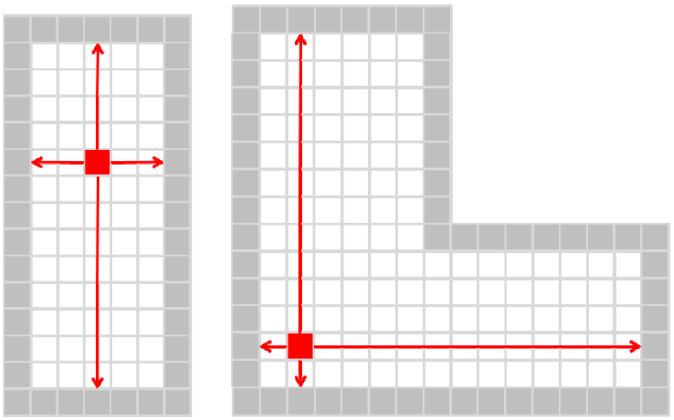


Figure 3.1: A sample of the SWT implementation taken from Epshtein et al. [Epshtein2010]. Darker pixels represent a stroke (a). From the green boundary pixel p (b), in direction of the gradient, we hit the red pixel q on the other end of the stroke. In between them, the minimum of either the width of the stroke, or the currently found width value, is assigned to each pixel (c).

SWT algorithm returns a image of the same size as the input image, in which each pixel has the value of stroke width associated with it. The first step to find strokes is using a canny edge detection [Canny1986] on the image. As shown in figure 3.1(b), the next step is to calculate a gradient direction for each edge pixel. For all boundary pixels, the gradient direction must be a normal to the stroke's orientation. When following the ray produced by the pixel plus the gradient direction's normal (see Figure 3.1(a)), an edge pixel on the other side of the stroke is found. If there is no edge pixel found on the other side of the stroke, the algorithm discards the ray and continues to the next edge pixel. Otherwise the stroke's width, assigned to each pixel along the ray, is defined as the length between the two boundary pixels. However, if a pixel already has a width value assigned to it, the minimum

of the two values is chosen as can be seen in figure 3.2(a). Some complex pixel candidates,



(a) (b)

Figure 3.2: A stroke width assignment example of a pixel, taken from Epshtein et al. [Epshtein2010]. A correct pixel assignment (a), of the minimum stroke width value of the vertical and horizontal lengths of rays between boundary pixels. A more complex example (b), shows an incorrect assignment of the minimum stroke width value between two rays. In this second example, the correct stroke width would be the length between the pixel and the corner. Therefore, a second median SWT pass is needed for more complex candidates.

such as corners, will be assigned with false stroke width values as shown in figure 3.2(b). Hence, a second median SWT pass is computed for every non-discarded ray. In this second pass the median stroke width value is computed for all the pixels of a ray. All pixels assigned with a higher stroke width value than the median are then set to the median value.

Finding suitable letter candidates:
As a basis for finding suitable letter candidates out of the SWT image, Epshtein et al. modified the connected component approach of [Horn1986]. A sample of a SWT image is shown in figure 3.3. They changed the association rule to compare stroke width values of pixels, instead of a binary mask. The algorithm was applied for the positive and negative gradient direction to enable both dark text on bright background and bright text on dark background. A letter can contain several neighbouring pixels if their stroke width values are alike. Epshtein et al. used a small stroke width ratio as a threshold for the grouping. This also allowed a pixel grouping for strokes which widths changed slightly, as in more complex fonts.
After grouping neighbouring pixels, they filtered these potential letter candidates according to a set of rules. The following list summarizes the rule set used to filter the letter candidates:

(a) (b)

Figure 3.3: The SWT image (b) computed for a sample input image (a).

- Eliminate connected components which have a stroke width variance that is to high. This rule eliminates foliage, for example, which is known to be a common false positive candidate of text detectors. And since text regions are more consistent than foliage regions this rule works, according to [Epshtein2010].
- Limit the connected component's aspect ratio to be between 0.1 and 10. Furthermore, limit the ratio between the component's median stroke width value and their diameter to be less than 10. Very narrow or long false positive components are eliminated by this rule.
- Eliminate components, which are smaller than 10 pixels or larger than 300 pixels. This filters a lot of aliasing from the canny edge detection.
- The maximum number of components inside the bounding box of another component is set to two. This ignores forms that surround text, such as sign frames.

All connected components, which pass this set of rules, are regarded as letter candidates.

Aggregating text lines:
To further improve their algorithm, Epshtein et al. grouped the extracted letter components to sets of letters. They expect text to be linear and to have similar characteristics such as size, average color, or stroke width. Also two letters must be within a distance of three times their width to be connected to the same word. Epshtein et al. state that all the thresholds for the letter rule sets and the word filter parameters were learned on a trainings set of the ICDAR text database.
Following these filtering steps, the algorithm further clustered pairs of letters into word chains. These chains are merged several times, until no more neighboring chains with a

Algorithm	Precision	Recall	f	Time(sec.)
Epshtein	**0.73**	**0.60**	**0.66**	**0.94**
Hinnerk Becker	0.62	0.67	0.62	14.4
Alex Chen	0.60	0.60	0.58	0.35
Qiang Zhu	0.33	0.40	0.33	1.6
Jisoo Kim	0.22	0.28	0.22	2.2
Nobuo Ezaki	0.18	0.36	0.22	2.8
Ashida	0.55	0.46	0.50	8.7
HWDavid	0.44	0.46	0.45	0.3
Wolf	0.30	0.44	0.35	17.0
Todoran	0.19	0.18	0.18	0.3
Full	0.1	0.06	0.08	0.2

Table 3.1: A performance comparison of text detection algorithms at the ICDAR 2003 and 2005 competitions, adopted from [Epshtein2010].

similar direction can be found.
To compare their results with the ones on the ICDAR database, they applied one last step. They used a heuristic, which uses a histogram of horizontal distances between neighboring letters and computes a threshold to separate text lines. This heuristic is used to partition the prior merged chains into the final text lines.

The SWT algorithm of Epshtein et al. [Epshtein2010] outperformed many other state-of-the-art text detection algorithms when we started this thesis. They published a precision and recall performance comparison based on the ICDAR text detection competitions (2003 and 2005), were they performed 73% in precision and 60% in recall, resulting in a 66% combined measure of quality (4% better than the second best algorithm). The full comparison list, published in [Epshtein2010], can be seen in table 3.1. To our knowledge, it was then the best performing state-of-the-art algorithm. We therefore decided to use the SWT algorithm in our thesis and add it as a saliency feature to our computational attention model (see chapter 5). As stated above however the method of computing text saliency features for this computational attention model does not change when the text detection algorithm is exchanged.
There is, no published source code for the SWT algorithm provided by the authors. To implement the SWT algorithm we used the CCV library[2], which provides a fairly good implementation. The CCV approach implements the above described algorithm, but lacks some parameter tuning and the correct handling of the resulting word detection mask. With a non-modified version of the SWT implementation of the CCV library, we only managed to achieve a score of 58% precision and 60% recall. To reach comparatively good precision and recall results as proposed by Epshtein et al. we had to modify the CCV li-

[2]http://libccv.org/

brary's source code and parameters. To use the resulting text areas as saliency features in our model, we also had to implement a saliency representation of the detected text areas. A further description of the modified CCV's library SWT algorithm, and our use of it in the attention model, can be found in section 5.1.2.

3.3 OpenCV scene text detector

As stated above, we wanted to compare the performance of a computational attention model using the SWT approach for text features to a more recent text detection algorithm. Therefore, we chose to use the new scene text detection implementation provided by the OpenCV library. This implementation is based on two publications: the scene text detection algorithm by Neumann and Matas [NeumannMatas2012] and the the multi-script text extraction from natural scenes algorithm by Gomez and Karatzas [GomezKaratzas2013]. Neumann and Matas approach is based on the MSER algorithm, which is mentioned above. They selected suitable extremal regions (ERs) using classifier trained for character detection from the entire component tree of an image. This selection is carried out sequentially to improve the computing time. The classification is done in two phases:

Compute Incrementally Computable Descriptors such as area, bounding box, perimeter, euler number and horizontal crossing for every region. These descriptors are then used as classifier features to compute the probability that the region is a character. Hence, only ERs which are classified as potential character regions are selected for the second phase.

Compute Sequential Classifiers such as the hole area ration, the convex hull ratio and the number of outer boundary inflexion points, to further classify selected potential character regions.

This selection process is computed in several projections to increase a characters recall.

The grouping of the selected potential character regions into high-level text blocks is then achieved with the approach of Gomez and Karatzas. They proposed a method to find reasonable groups of characters with the use of a perceptual organisation framework that exploits collaboration of proximity and similarity rules. Their method used two clustering algorithms without parameters to detect potential groups of text regions. The first method finds the maximally reasonable groups within several sets of feature spaces that hold similarity measures and proximity parameters. Then they created hypotheses for potential groups of text areas for the resulting sets of feature spaces and used an evidence accumulation algorithm to combine them. These combined hypotheses of potential text groups are finally checked for their alignment since the grouping algorithm can only handle horizontally aligned text.

Figure 3.4 shows some sample results of the used text detection algorithms.

Figure 3.4: Sample results of the used text detection implementations. The left row (red bounding boxes) shows sample results computed with the SWT algorithm, the right row (yellow bounding boxes) shows sample results computed with the OpenCV scene text detector.

As can be seen by the performance comparison on table 3.1, and the performance results of [Lee2011] stated in section 3.1, the problem of detecting text in natural image scenes is far from solved. This means, of course, that a text detection feature used in a computational attention model will miss some text regions or falsely specify non-text regions as text. With this knowledge, and since there was no better performing solutions available, a quality measure of 66% as performed by Epshtein et al., provides a good basis for text specific attention models. Furthermore, we compared the computed model using Epshtein et al.'s approach to a second text detection algorithm implemented by the OpenCV library to show how the used text detection algorithms change the performance.

Chapter 4

Visual attention study

In this chapter we are going to describe the user study we did for our attention model. This study recorded human fixations on an image database, using a remote eye-tracking device. Section 4.1 will be on why we needed to do a user study, followed by a description of the used database in section 4.2. In Section 4.3 we will explain our eye tracking setup as well as provide an in-depth summary on the recording procedure. And finally in section 4.4 we explain which algorithms we used to transform raw eye data to fixations.

4.1 Purpose of the study

Since this thesis is based on the model of Judd et al. [Judd2009], we also wanted to be as comparable as possible to their approach. Judd et al. did an eye-tracking user study to gather ground truth data for their model, using the MIT database[Judd2009]. After a thorough analysis of the stimuli images of this database, we found that only 7.9% of the images contain scene text. We, however, wanted to improve Judd et al.'s model with a text detection component. Therefore, we decided to add a database, which focuses on scene text, to the MIT database for our study. We chose about 500 images from the ICDAR[Lucas2003] dataset, containing scene text. To gather the new human fixations that were needed to retrain our computational attention model, we carried out a new eye tracking study containing both of these datasets. An eye tracking study only containing the ICDAR images would not have been sufficient since we would have had mixed gaze data from our participants with those provided by Judd et al. To provide clean and continuous eye tracking data a used database should contain the eye tracking data of the same participants for each image. The alternative to this combined dataset would have been to collect completely new stimuli images containing both natural scene images as well as scene text images. We did however refrain from this option since Judd et al. already showed good results with a model trained on their collected natural scene context, and we wanted to compare the influence that additional text context might provide to this general dataset.

As stated above, we wanted this study to be as similar as possible to the one Judd et al. did in [Judd2009]. Therefore, we will shortly summarize their data gathering protocol

first. They had fifteen participants between 18 and 35, two of them were researchers on the project, the others naive viewers. All of them were sat approximately 60 cm apart from a 19 inch screen, with a resolution of 1280x1024. The company or model of their eye tracking device is not stated in their approach, however they used a chin rest to stabilize the head. We also know that the sampling rate of their eye tracker was 250hz. Each participant free viewed the 1003 images of their MIT database, in a random order. To avoid the loss of a participant's concentration, they divided the viewing into two sessions. At the beginning of each session the eye tracking device was calibrated, and a validation process was done for every 50 images. Validation means that the device's calibration is checked for correctness and re-calibrated if a non tolerable offset is reached. Each image was shown at full resolution for 3 seconds, followed by a gray image containing a cross-hair at the center for 1 second. The cross-hair at the center is a standard method to refocus the viewer's attention to the center of the screen before presenting the next image. Furthermore, the carried out a memory test after each session, showing the participants a couple of images and asking them if they had just seen these images. These memory tests were done to motivate the participants to pay attention. We were not able to gather any more details on the data gathering protocol from the published approach [Judd2009].

4.2 Image database

Judd et al. [Judd2009] already used an extensive image database for their user study, called the MIT database[1]. This database contains 1003 stimuli images they collected randomly from flickr creative commons[2] and LabelMe [Russell2008]. The image dimensions of these images were 1024 pixels on the larger side and varied from 405 to 1024 pixels on the smaller one. It was created to provide ground truth data for computational attention models and for large scale quantitative analysis of eye movements. Judd et al. provided the image database, all collected eye tracking data, as well as the Matlab source code they used to collect those on their web page.
As already stated above, even though this database already has an extensive amount of stimuli images, it only contains 79 images with visible scene text (7.9%). And only 32 images, which are **3.2%** of the entire database, contain text that is not far too small, occluded, blurred or possibly undetectable for current state-of-the-art text detection algorithm in any other way. Therefore, as stated above, we decided to add a database of natural scene text stimuli images to the existing MIT database. We chose the ICDAR[3] database [Lucas2003] which was proposed as a training, as well as a validating, database for the robust reading competitions for ICDAR 2003. Lucas et al. [Lucas2003] provided a common benchmark dataset, to compare the performance of text recognition algorithms in natural scenes. They provided train and validation image datasets for three sub problems:

[1] http://people.csail.mit.edu/tjudd/WherePeopleLook/index.html
[2] http://www.flickr.com/creativecommons/
[3] http://algoval.essex.ac.uk/icdar/Datasets.html

text location, character recognition and word recognition. We decided to use the combined trainings and validation image sets of the text location dataset, because these provide a large amount of scene text in natural environments. Therefore, we chose 250 images of the trainings set and 245 of the validation set, leading to a database of 495 scene text stimuli images.
With the MIT and the ICDAR datasets combined we gathered a database of 1498 stimuli images, where 38.5% of the images contained scene text. An extended database meant that we needed to redo an eye tracking user study to gather new ground truth gaze data.

4.3 Eye tracker setup and data gathering

Figure 4.1 shows a sample of our eye tracker setup. In our study we used a SMI RED500[4]

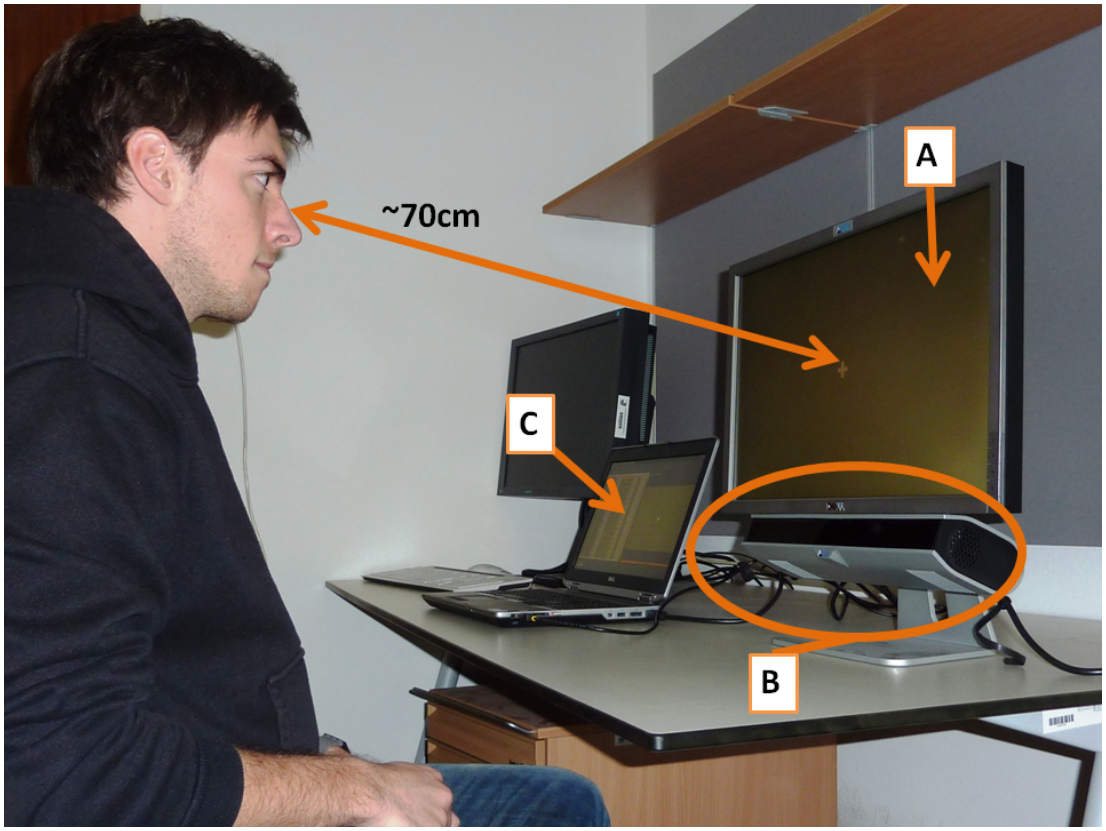


Figure 4.1: The study's eye tracker setup. Participants sat about 70cm apart from the 22"stimuli screen (A) that showed stimuli images in full screen, at a resolution of 1680x1050. A SMI Red500 eye tracking device (B) collected eye movement data at a sampling rate of 250hz. While all data was viewed and stored on the supervisors laptop (C).

remote eye tracker, which is able to gather gaze data with up to 500hz. The RED500 is a remote eye tracker that collects binocular gaze and pupil data. It allows free head movement (40cm x 20cm at a distance of 70cm), and several calibration modes (2, 5 or 9 point calibration). According to the manufacturer, it also works with most glasses and contact lenses. However, as described in [Holmqvist2011] glasses and contact lenses can still be a problem if an experiment has to be very accurate. We tried therefore to acquire participants with normal eye sight. To stay comparable to Judd et al. we decided to record our data using 250hz instead of 500hz. We placed our participants about 70 cm apart from our 22"computer screen that showed all stimuli images to the participants. The screen's resolution was 1680x1050, and we showed the images in full size.

Data gathering protocol:
The study for this thesis was also done with fifteen participants. Eight female and seven male participants, aged between 18 and 33, with a mean age of 25.13 and a standard deviation of 3.7 years, attended the study. All of the participants had normal or corrected-to-normal eye sight. Furthermore, all of them were naive to the task, which is important because participants with prior knowledge of the project's focus on text would probably be more prone to look at scene text.
Larger images from the ICDAR database were down-scaled before the study, to fit the screen, so we could still used a full-size view for all images. Since we added 495 images from the ICDAR scene text database to the MIT database, we had a total number of 1498 images to show each user.
We used the same viewing setup as Judd et al. with 3 seconds per image, each followed by a gray image, containing a cross in the center, for 1 second. With this setting, a session would take one hour and 40 minutes only for the viewing task. Including calibration at the start, validation for every 50 images and the memory test at the end, each participant would have to be concentrating for more than two hours. This is why we chose to partition the study into three sessions for each user. The first two sessions containing 500 images each and the last 498. For each session the eye tracking device had to be calibrated. We did a 9-point calibration, and also validated the calibration every 50 images. At the end of each session we also carried out a memory test, which participants had been told about in advance. Each participant got reimbursed 20 € per session for their time.

Validation process:
The validation process, provided by the SMI recording software we used to collect the eye data, calculated the so-called gaze *accuracy*. Accuracy, as described in [Holmqvist2011], is defined as the average difference between the recorded gaze position and the true gaze position. The accuracy measured by eye tracking software modules is commonly an average angular distance (offset) θ_i between the n recorded gaze locations and the defined location of the gaze targets. Holmqvist et al. stated that this is an operational definition

[4]http://www.smivision.com/en/gaze-and-eye-tracking-systems/products/red-red250-red-500.html

of accuracy, also called *offset*,

$$\theta_{Offset} = \frac{1}{n}\sum_{i=1}^{n}\theta_i \tag{4.1}$$

Holmqvist et al. further stated that, depending on the experiment, a reasonably small offset (therefore good accuracy) can span from 0.3° to 2°. There are several factors which may influence the accuracy during eye tracking recordings, such as glasses, contact lenses, varying eye physiologies, varying levels of lightning, strong head movements or problems while calibrating. They collected reported measurements from several experiments of different researches over a few years, which brought up different arguments as to why inaccuracy should be below certain values. A good accuracy offset for general remote eye tracking studies seems to be an average offset of $\leq 0.5°$, which equals 20.5 pixels at a distance of 70cm. Therefore, we decided to use this offset as a maximum deviation when validating, and re-calibrated the eye tracker when participants achieved higher inaccuracies during the experiment.

4.4 Transforming eye data to fixations

The data we collected with our eye tracking study was raw data that consisted of eye gaze position and time values for each sample. To gather the more useful fixation information, we used Torralba et al.'s [Torralba2006] acceleration based algorithm to separate the data samples into fixations and saccades. Torralba et al. defines saccades as a combination of distance and velocity criteria between eye movements. The acceleration criteria to detect saccades by distance d and velocity v is defined in equation 4.2.

$$\mathbf{d} = p * t; 0 <= \mathbf{v} <= n/r; \tag{4.2}$$

The pixel per degree (p) is based on the used eye tracking device, the degree per second is defined as an acceleration threshold of value 6, the number of samples (n) divided by the sampling rate (r) defines the maximum velocity time in milliseconds. Data samples with a smaller movement as defined by the criteria are considered a drift within a fixation, while data samples with greater movements are defined as saccades. Fixations have a fixation duration (the time span between two saccades) and a position that was defined as the average position of all data points of the fixation. We decided to use Torralba et al.'s method, again, to stay comparable with the procedure of Judd et al. [Judd2009]. The source code of this algorithm can be found on the MIT web page, provided in the adopted version by Judd et al.[5].
As stated above, it is a common procedure to refocus a viewer's attention to the center before a new stimuli appears. However, when evaluating the data samples it is also common to discard the first fixation from each stimuli for each user [Judd2011b]. This is done to eliminate the irrelevant initial center fixation information. As proposed by Judd et al.

[5]http://people.csail.mit.edu/tjudd/LowRes/Code/checkFixations.m

[Judd2009] we also used only the first six fixations of each user, since only the first few seconds of seeing a new stimulus are said to be important for human attention. Judd et al. [Judd2011b] also states that the first few fixations are commonly more consistent, either because they tend to look at the center first or they tend to look at more salient regions first. And therefore, early fixations are more easy to predict, according to Judd et al. Figure 4.2 shows sample images of the top six human fixations (4.2(e)-4.2(h)) for sample images from both databases (ICDAR 4.2(a), 4.2(b) and MIT 4.2(c) 4.2(d)).

(a) (b) (c) (d)

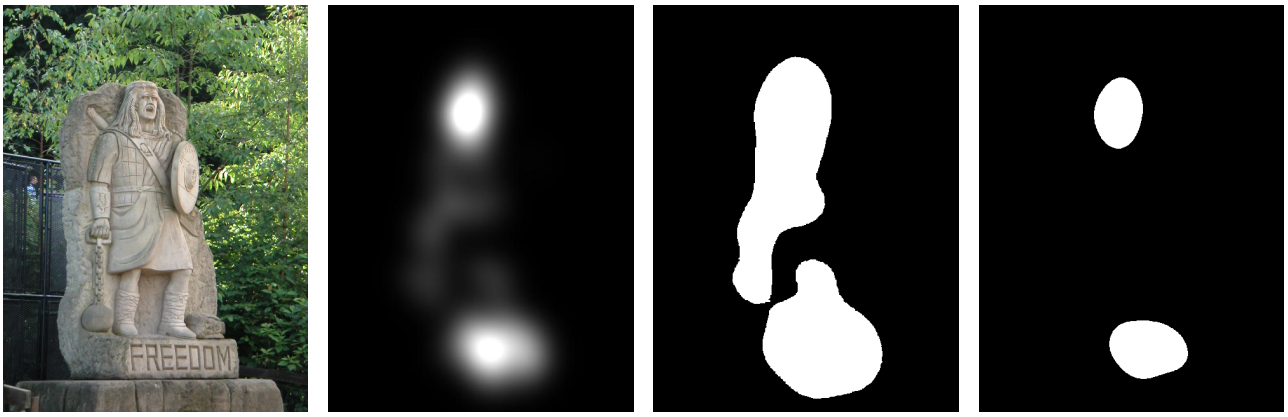
(e) (f) (g) (h)

(i) (j) (k) (l)

Figure 4.2: The top six human fixations for all 15 users (e-h), and computed fixation maps (i-l) for sample stimuli images from both, ICDAR (a,b) and MIT (c,d), databases.

Once we gathered this fixation information, we transformed it to a continuous saliency map for each image, as proposed by Judd et al. [Judd2009]. This was done by convolving a Gaussian filter across the top fixations of each viewer. We also computed such saliency maps for single participants and all but one of the participants for the experimental analysis (see chapter 6). Thresholding these saliency maps means that binary maps of the top $n\%$ saliency locations of the stimuli images can be aquired. Figure 4.3 shows a sample

image of human fixation saliency map thresholded to the top 20% and top 5% of saliency. Such thresholded saliency maps are what most attention model aided applications use



(a) (b) (c) (d)

Figure 4.3: Sample images of top percent saliency thresholded fixation maps. A full human fixation saliency map (b), for a sample stimuli image (a), convolved with a Gaussian filter across the top fixations of all participants is thresholded at the top 20% (c) and the top 5% (d) of saliency.

to improve their performance. However, the saliency maps we are explaining here are all created from real human fixations and not yet computed by any computational attention model. We need those human fixation saliency maps to train the machine learning part of our model and to use them as ground truth to validate the performance of the model. The basic source code for the above described computations was provided by Judd et al. on their project's web page[6]. Their method of creating human fixation saliency maps out of computed fixation data was not part of this source code. Since we used a different eye tracking device, resulting in different raw data, and not all of the source code used by Judd et al. was available, we had to modify and extend the source code to produce similar human fixation saliency maps.

The data quality of recorded eye gaze data is dependent on many factors. Holmqvist et al. [Holmqvist2011] give a good overview of the most important steps to be aware of before, while and after carrying out an eye tracking experiment. We noticed that Judd et al.'s experiment setup was planned quite well, since we tried to reproduce the study as closely as possible. What we learned from our study was that first of all: a repeated calibration validation step within a recording session is very important, because the accuracy can degrade over time, as described above. Furthermore, the transformation from raw data to fixation data, and further to continuous human fixation maps, was very tricky because small disparities would lead to completely different results.

[6]http://people.csail.mit.edu/tjudd/WherePeopleLook/Code/DatabaseCode.zip

To our knowledge, the eye tracking experiment for this thesis provided the largest collection of gaze data including general and scene text focused images. In the following chapter we will explain how we trained our computational attention model using the human eye tracking data described above.

Chapter 5

A new computational model of human visual attention

One of the contributions of this thesis is the extension of the saliency model by Judd et al. [Judd2009]. As described previously, we have proposed to add a state-of-the-art text detection approach [Epshtein2010] and using newly collected human fixation data (see chapter 4) to train a saliency classifier. Since the framework we have used is based on Judd et al. we are going to describe this approach in section 5.1. Further we will describe the image feature collection they used in section 5.1.1, how we added the text saliency approach in section 5.1.2 and finally how the SVM classification has been done in section 5.1.3.

5.1 The Framework

The main contribution of Judd et al.'s [Judd2009] approach was that instead of only combining bottom-up features and biologically plausible filters, their saliency model learned a saliency classifier from human fixations. In this section we are giving an in-depth description of Judd et al.'s framework, which features are used and how the learning process has been carried out. The framework we are working with is a supervised learning model of saliency that combines bottom-up cues with top-down semantic depended cues. Judd et al. did an eye-tracking user study on an extensive image database, which they used as ground truth data to feed the machine learning algorithm and also to validate the trained model. For more details see chapter 4, or the data gathering protocol in section 4.1. The basic work-flow of their approach is shown in figure 5.1 and shortly summarized in the following:

Collect a data-set of images and corresponding human fixations for each image. Then partition these images in a set of training images and a set of test images.

Compute all features for each training image, and store them as a sub-feature matrix, where each column represents the feature vector of one image.

Select positive and negative samples for each training image. These are randomly cho-

sen pixels from the top p percent of the ground truth data for the positive samples, and bottom q percent for the negative samples. Each of these pixels is assigned a label (*1* for positive samples and *0* for negative samples) and stored in a label vector that is subsequently used by the machine learning algorithm.

Create a feature matrix out of the chosen sample pixels, from each prior calculated feature vectors. This reduces the data size extensively, depending on the parameters chosen for p and q as well as the number of different image features.

Whiten the data to ensure zero mean and unit variance. Since the image features may be computed in different data ranges we need to adapt them to ensure useful classifications between the image features.

Learn the saliency classifier by using a linear support vector machine (SVM). The positive and negative training samples for the SVM are provided from the prior defined feature matrix, and the labels associating pixel values with their classification (positive or negative), are provided by the label vector. As a result the classifier produces a weighting value for each image feature.

The Matlab source code of this basic framework, the stimuli database and the human fixation maps of the [Judd2009] publication are all available for download on their webpage[1]. We will describe the important tasks of this work-flow in the following sections in more detail as well as what modifications and extensions we carried out.

5.1.1 Feature collection

The concept that a combination of pre-attentive feature detection mechanisms lead to an overall saliency map, which provides efficient strategies for deploying attention on the basis of bottom-up cues, was already introduced by Koch and Ullman [KochUllman1985] (see section 2.1.1). As stated in chapter 2, Judd et al. [Judd2009] used several low, mid and high-level image features to define salient locations of input images. When using Judd et al.'s Matlab framework we first had to install several feature detector libraries to be able to carry out the feature collection step. Figure 5.2 shows Judd et al.'s 33 combined features (including sub-channels), which we will now describe in more detail:

Low-level features: were used because they have been shown to provide good predictions of human visual attention and are physiologically plausible [Judd2009]. Judd et al. used the local energy from steerable pyramids **subband features** [SimoncelliFreeman1995] in three scales and four orientations (figure 5.2 images 3 -15). A steerable pyramid is defined as a multi-scale/-orientation image decomposition in the frequency domain, which allows efficient representations independent of orientation and scale. An image is first partitioned into lowpass and highpass subbands, whereupon the lowpass subbands are then divided into several oriented bandpass subbands and further subsampled for x and y directions. These subbands are translation-invariant and

[1] http://people.csail.mit.edu/tjudd/WherePeopleLook/index.html

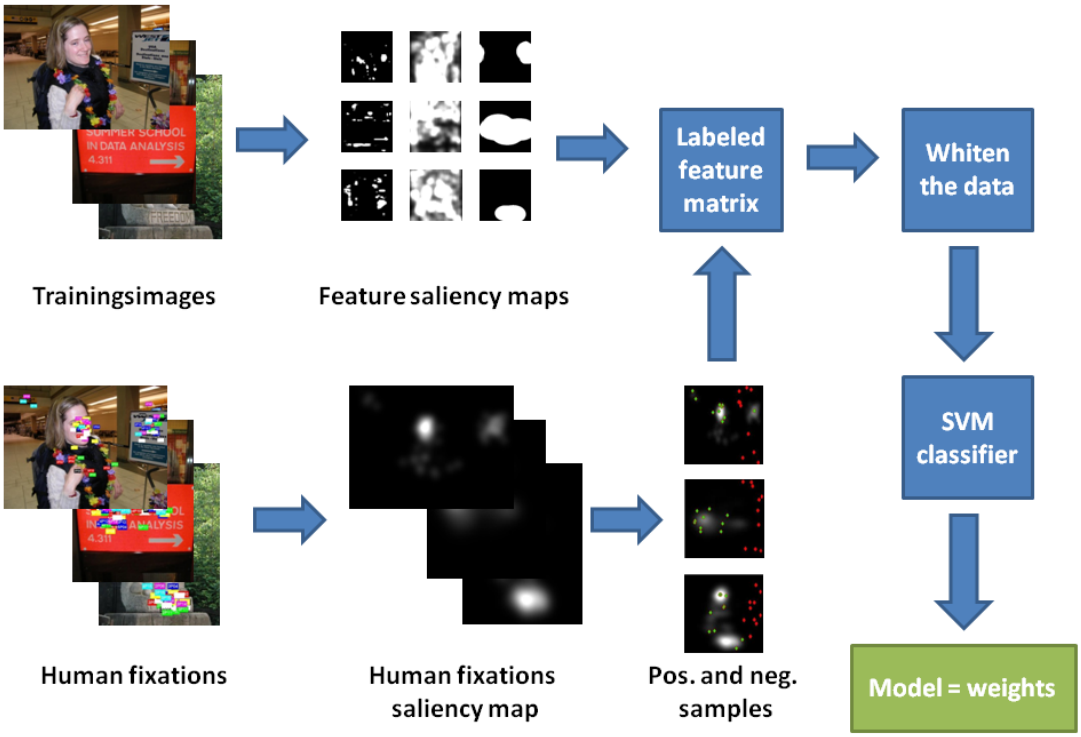


Figure 5.1: A simplified diagram of the computational model's work-flow. Saliency maps for each image feature are calculated for each training image. Positive (green dots) and negative (red dots) samples are randomly chosen from the top p and bottom q percent of the human fixation saliency maps, which are created from actual human fixations (see section 4.4). A labeled feature matrix is then created from the selected samples of each feature saliency map and is further whitened to ensure data coherence between the different features. This matrix is then used as input for a SVM classifier, which provides a so-called model containing a weight value for each image feature.

rotation-invariant and are constructed into a pyramid hierarchy. Judd et al. used Simoncelli and Freeman's [SimoncelliFreeman1995] approach, which can be downloaded from their webpage[2], to find the steerable pyramid subbands, and blurred each band with a Gaussian filter to use them as saliency maps. They also included the so-called **Torralba saliency feature** based on the simple saliency model approaches of [Rosenholtz1999] and [OliviaTorralba2001] (figure 5.2 image 30). This simple model computes saliency values from probability computations of different histogram layers of the subband pyramids, multiplied by a weighting value for each

[2]http://www.cns.nyu.edu/~eero/steerpyr/

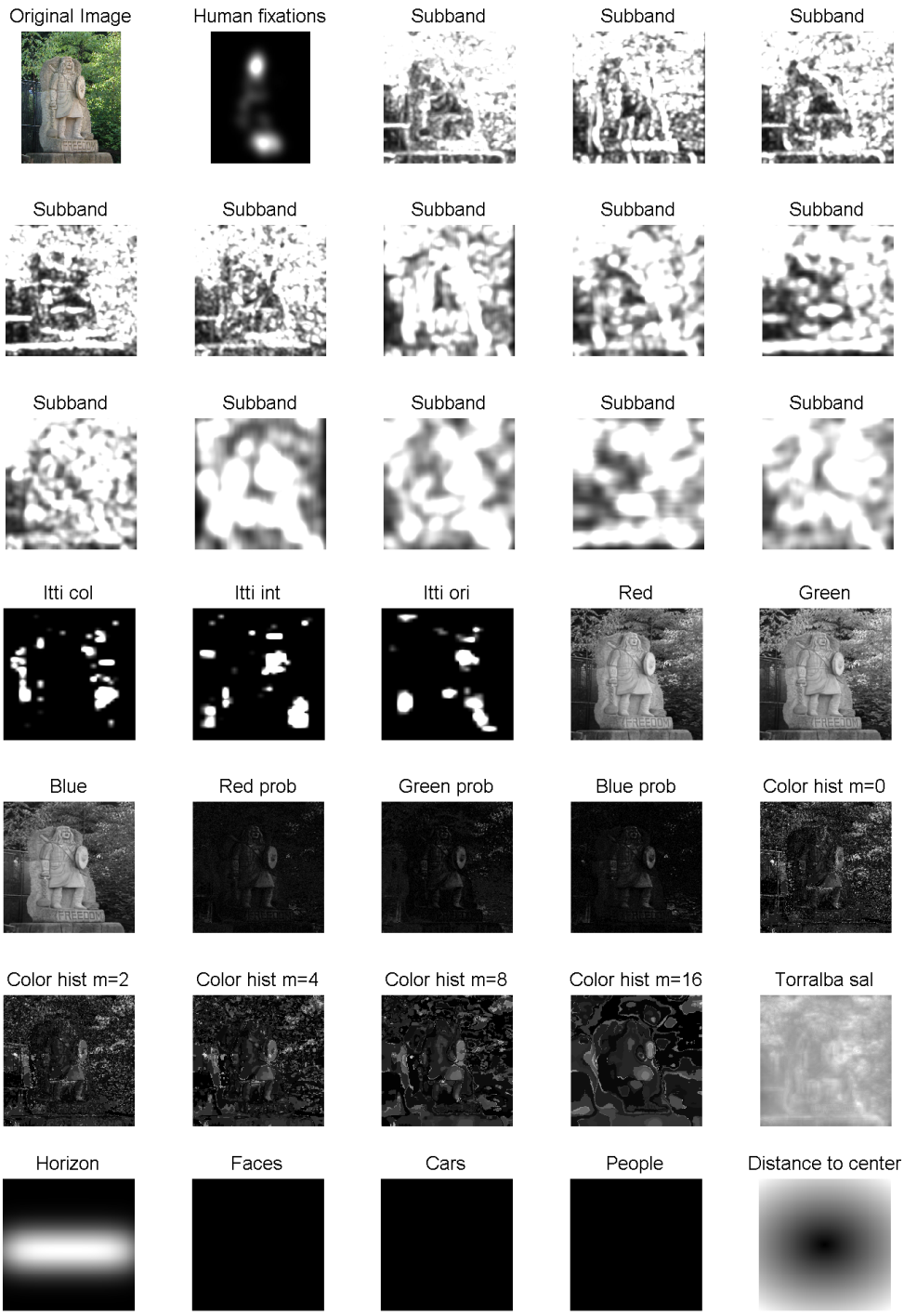


Figure 5.2: Judd et al.'s [Judd2009] features. A sample of all 33 image features computed for one sample image (top left) and the human fixations (second image).

scale. As stated in chapter 2 it is well-known that intensity, color contrast and orientation are important bottom-up features, as shown for example in Itti et al.'s *iLab Neuromorphic Vision C++ Toolkit* [Itti1998]. Therefore, Judd et al. included these three feature channels (figure 5.2 images 16-18) as **Itti and Koch saliency features**, using the Saliency-ToolBox (STB)[3] [WaltherKoch2006] described in section 2.1.1. Since color is an important feature for natural scene image based attention modeling, they further included several **color feature** channels such as red, green, blue (figure 5.2 images 19-21), the probabilities of these channels (figure 5.2 images 22-24), as well as the median filtered probability from a 3D color histogram per color channel at six scales (figure 5.2 images 25-29).

Mid-level features: as Judd et al. state in their approach most objects in a scene rest on earth's surface, which is why humans naturally fixate upon the **horizon** (figure 5.2 image 31). Therefore, they used mid-level gist features as described by [TorralbaSinha2001] to detect the location of the horizon line on a scene image. The algorithm to detect horizon lines can be downloaded as part of the LabelMe toolbox[4]. They further blurred the horizon line, providing a continuous saliency map of the attentive horizon area.

High-level features: As stated in chapter 2, objects, especially faces, are image features to whom the human attention is commonly drawn (see section 2.1.2 for reference). Therefore, Judd et al. included Viola and Jones's [ViolaJones2002] **face** detector (figure 5.2 image 32) and Felzenszwalb et al.'s [Felzenszwalb2008] **car** and **people** detector (figure 5.2 images 33-34). Judd et al. used a Matlab implementation[5] of the face detector by Viola and Jones and the Matlab code Felzenszwalb et al. provided on their webpage[6]. These algorithms were state-of-the-art object detection methods at that time and we decided not to modify them since our focus was including text saliency to the attention model.

Distance to center feature: is an image feature with the purpose of reducing the natural center bias, which occurs when humans commonly frame an interesting object at the center of an image. As stated in section 2.1.2 the center bias is an important aspect of human visual attention, but was only included in a few computational attention models [Judd2009], [ParkhurstNiebur2003] and [ZhaoKoch2011]. Therefore, Judd et al. included a **center feature** (figure 5.2 image 35) that computed the distance to the center for each image pixel.

We reused these feature detectors, and set up the basic feature collection algorithm proposed by Judd et al. [Judd2009]. The newly applied text saliency feature from this thesis will be described in the next section. All of these computed features were resized to

[3] http://www.saliencytoolbox.net/
[4] http://labelme.csail.mit.edu/LabelMeToolbox/index.html
[5] http://www.mathworks.com/matlabcentral/fileexchange/19912-open-cv-viola-jones-face-detection-in-matlab
[6] http://people.cs.uchicago.edu/~pff/latent/

a dimension of $200x200$ to improve computation time.
The next steps of the learning process will be described in section 5.1.3.

5.1.2 Adding the text feature

As stated in chapter 3, it does not matter which text detector we include with the computational attention model . However, based on the hypothesis that a text feature will improve the model's performance, we expect that the results will increase the performance when we choose better text detectors. To demonstrate the impact of using different text detectors, we decided to include a newer text detection approach for comparison. We will first show how we computed the saliency map out of detected text areas for each image and then elaborate further on the differences between used text detectors.
Since a saliency map is the representation of a certain interesting feature within a scene image, it is usually a grayscale image, where brighter values are assigned to more salient pixels. For some features a binary saliency map (either a pixel is salient or its not) is sufficient, such as the face detection feature where the detection bounding box lies within the face. Other features need a more differential measurement and usually represent the likelihood or weight of a salient pixel (again brighter means more salient), such as the colorspace features, subband features, distance to center feature and model based features like the Torralba saliency feature. Text detection algorithms provide a rectangular bounding box for text regions in a scene image and one possible approach for creating text saliency images would therefore be to create a binary map setting all pixels inside these text bounding boxes as salient pixels. However, due to the natural properties of characters, these bounding boxes often hold more background pixels than actual character pixels, in contrast to, for example, face or car detection algorithms. To separate character pixels from background pixels, a text segmentation that produces a binary map of black background and white character pixels is needed. Using those text segmentation results directly as text saliency maps could be another approach, however, the sample scan path of a study participant reading the scene text in figure 5.3 shows that the human fixations do not necessarily rest directly on a character pixel even when we look at the text. Therefore, we decided to blur the resulting text segmentation, using the same Gaussian blurring algorithm Judd et al. [Judd2009] used to blur their human fixation maps, to create a continuous grayscale text saliency map. This third possibility is, to our knowledge, a novel approach to creating text saliency images and an experimental comparison between different models trained with the three possible approaches showed that this third approach yields the best prediction results. We will now further describe the work-flow we used to produce the text saliency maps that is shown in figure 5.4:

Text detection: The SWT [Epshtein2010] text detection algorithm we used for the scene text detection is already described in detail in chapter 3. For the text saliency work-flow we need this algorithm to gather positions of potential text regions in a given input image. We created a binary of the CCV library SWT algorithm implementation (see section 3.2) that could be called as a system call from the Matlab implementa-

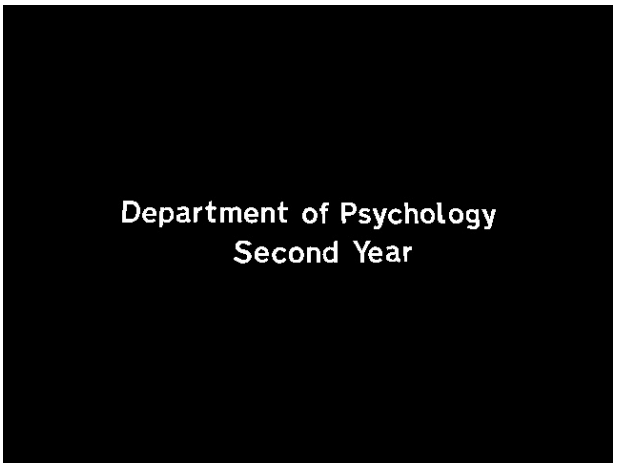
Figure 5.3: A sample scan path of one of our participants shown for the text segmentation image. Fixation points are shown as yellow dots, saccade data points as red dots, fixation labels are starting at the second fixation (see section 4.4 for clarification). As shown in this sample, human fixations don't always rest exactly on the letters when we read, therefore, a blurring as in figure 5.4(c) is applied to the text segmentation in order to achieve correct behavior when training a classifier with random human fixation samples.

tion of the model's feature collection, as described above. The created SWT binary requires the input image as well as threshold parameters for the canny edge detector, all other SWT relevant parameters were hardcoded into the source code. As visualized in figure 5.4(a), the algorithm returns the pixel-coordinates of a rectangular bounding box for each text region found for a given image. This part of the work-flow is the most important regarding detection performance, since the other parts of the work-flow are dependent on the precision of the text detection results.
However, as stated before, this part is also replaceable with other text detection algorithms which may improve the performance. To show this we did an experiment with another text detector by exchanging the SWT module with the scene text implementation of the *OpenCV library*[7]. Their implementation is based on the scene text algorithm proposed by [NeumannMatas2012] for detecting characters candidates using Extremal Regions (ERs). For the grouping of these character candidates into text blocks, the OpenCV implementation uses Gomez and Karatzas's [GomezKaratzas2013] approach. We will further call this approach based on the two publications, which both have already been described in chapter 3, *OpenCV text detector* (OCVTD). We developed a simple C++ scene text detector using this OCVTD implementation that was also called from the feature collection script and also returned a rectangular bounding box for each text area of a given image. This made it easy to exchange the OCVTD with the SWT text detector and vice versa.

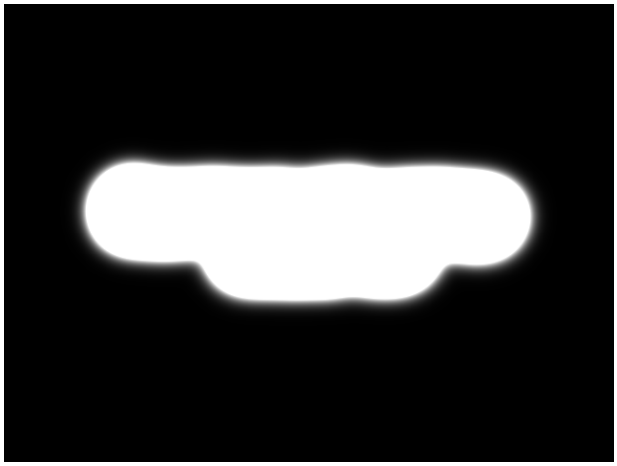
We did an adapted precision and recall evaluation on the ICDAR test dataset to

[7]http://opencv.org/

(a) Text detection



(b) Text segmentation



(c) Text saliency

Figure 5.4: The text saliency image creation process: first we use the text detection algorithm (a) to gather text regions in the original image, next we use the text segmentation approach (b) to separate characters from background and last we blur the segmentation result to get a continuous saliency map (c) of the image's text regions.

compare the detection performance of both algorithms. The dataset was available on the ICDAR competition's webpage[8], where we chose the text location test dataset that contained the ground truth text locations for each image. Our text detection algorithms usually grouped text lines together and returned one estimation bounding box per text line E, the ground truth's text locations T on the other hand are stored as one bounding box rectangle per word. We marked the areas of intersections

[8]http://algoval.essex.ac.uk/icdar/Datasets.html#Robust%20Reading.html

Algorithm	Precision	Recall	f
OCVTD	0.81	0.66	0.72
SWT	0.73	0.62	0.67

Table 5.1: Performance comparison of the used text detection algorithms.

between E and T which had at least a 50% intersection area compared to the T rectangle area. The best match for a rectangle $r_e \in E$ and a $r_t \in T$ is defined as the $r\prime$ with the smallest difference of the area of intersection and the area of r_t. If we found more than one r_t rectangles inside one r_e as mentioned above, we summed the intersecting areas of all r_t up, checked if the summed up area matched to above criteria and if so we counted the r_e as number of $r\prime_t$ correctly matched rectangles. All correctly matched r_e according to this criteria were stored as true positives *tp*, all remaining r_t rectangles were counted as false negatives *fn* and all remaining r_e rectangles as false positives *fp*.

With this matching definition we computed precision and recall as:

$$precision = \frac{\sum tp}{|E|} \tag{5.1}$$

$$recall = \frac{\sum tp}{|T|} \tag{5.2}$$

The standard f measure that is used as a single quality measure from both precision and recall is defined as:

$$f = \frac{1}{\frac{\alpha}{precision} + \frac{1-\alpha}{recall}} \tag{5.3}$$

The weighting parameter α is set to 0.5 to give both precision and recall equal weights.

Our performance comparison of the two used text detection algorithms that were evaluated on the ICDAR 2003 test dataset is shown in table 5.1.

Text segmentation: For the segmentation we tested several available algorithms, of which an approach based on the well-known Maximally stable extremal regions (MSER) approach proposed by Matas et al. [Matas2004] and the segmentation approach from [Kasar2007], performed the best. The algorithm extracts co-variant regions (the MSERs) from a given image that represent a stable connected component for a certain level set of the image. Several text detection and localization algorithms use MSER-based approaches, such as Neumann and Matas [NeumannMatas2011], [NeumannMatas2012] for example, because well-selected sets of ERs can be used to detect characters in scene images. The MSER approach has previously been used for image segmentation, for example by Oh et al [Oh2013] who's algorithm collects MSERs and segments the image by drawing them in a specific order. Since we had

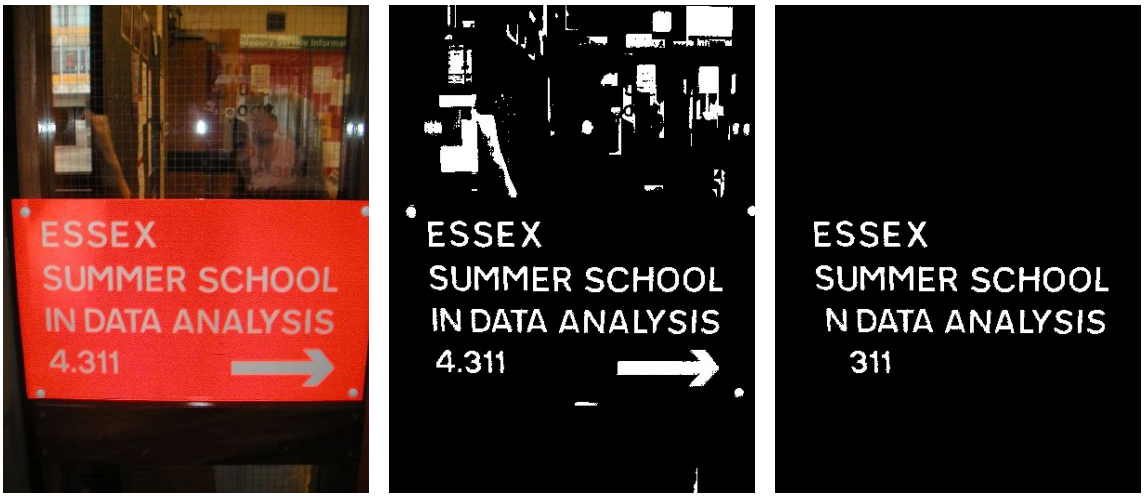
already collected the text detection bounding boxes, we applied a simple text segmentation within the found rectangular bounding box regions to separate the scene text from the background. We used the MSER implementation of a computer vision Matlab library called *VLFeat*[9] to collect the MSERs within these regions. The idea was to collect ERs for each character, which should be found more easily within only the bounding box of found text regions. We further exploited a drawing function from this library to draw the extracted boundary pixels of the ERs and all pixels within them on a blank image to the positions they have been found in the original image.
Kasar et al.'s [Kasar2007] Font and Background Color Independent Text Binarization approach provides an edge based connected component algorithm that determines the threshold for each component automatically. Their approach achieved good results on images with different background shades and multi-colored text in varying text sizes. We provided a C++ implementation of this approach that computes a binary image with white text character pixel for a given input image and a corresponding text file containing found text rectangles. We tested separately trained models including these approaches (after the text saliency step described in the following) and concluded that Kasar et al.'s approach yielded roughly 1% improvement, we therefore based the resulting tests in chapter 6 on this approach. Figure 5.5 shows our implementation of Kasar et al.'s text segmentation approach once applied on the entire image (figure 5.5(b)) and once applied only within the detected text locations (figure 5.5(c)) as used in our the text saliency process. As can be seen even tough the input image of figure 5.5(a) poses a difficult example for text segmentation (many small edge areas, reflective backgrounds and so on), if applied only on areas where text has supposedly been found it provides good results.

Text saliency: As stated above, since humans don't always fixate exactly on characters (as shown in figure 5.3) when reading, and we furthermore did not use all fixation data points for the model's training, we needed to blur the text segmentation in order to vote fixation data points that lie near characters as fixated on text. Using the proposed blurring algorithm votes the highest salient values directly on the characters and continuously less salient values the more distant a pixel is located to the nearest character pixel. We further normalized the blurred saliency images yielding maximum values of 255 directly located on the characters and 0 for pixels outside the text area rectangle.

Figure 5.6 shows some examples of the resulting text detection methods (first two rows) text segmentation method (third and fourth row) and the text saliency method (fifth and sixth row).

The model's feature gathering process was done as before, but we added the above described text saliency map as the 34th feature. Each feature of every input image was resized to an image dimension d, in this case $200x200$ and then transformed in a 1-D vec-

[9] http://www.vlfeat.org



(a) (b) (c)

Figure 5.5: The text segmentation approach of Kasar et al. [Kasar2007] on a sample image (a), once computed for the entire image (b) and once computed only within the detected text area locations (c) by the OCVTD algorithm.

tor resulting in a $40000x34$ feature sub-matrix per image. Each of these sub-matrices were subsequently combined to a MxN feature matrix, where M is the number of input images $n * d$ and N the number of features (34 for our model).

We created different models to compare their performance in our experiments, therefore we did the above described text saliency process for both text detection approaches and added each of them as the 34th feature to their own feature matrix. Furthermore, we remodeled Judd et al.'s approach with our new human fixation ground truth data gathered on our the combined dataset (see section 4.2), to compare their original approach to our adapted one.

5.1.3 Learning the attention model

The next step of the model's work-flow was to select positive and negative samples out of the feature matrix above described to generate a training set for the machine learning process. The positive and negative samples were randomly chosen pixels from the top p percent for the positive samples, and bottom q percent for the negative samples of the ground truth human fixation data. We chose 10 positive and 10 negative samples from each image, because according to Judd et al. [Judd2009] using more than 10 samples did not increase their model's performance. We also kept Judd et al.'s percent salient selection for p as the top 20% salient locations and q as the bottom 70% salient locations to acquire strongly salient and accordingly strongly not salient pixels. Furthermore, as Judd et al.

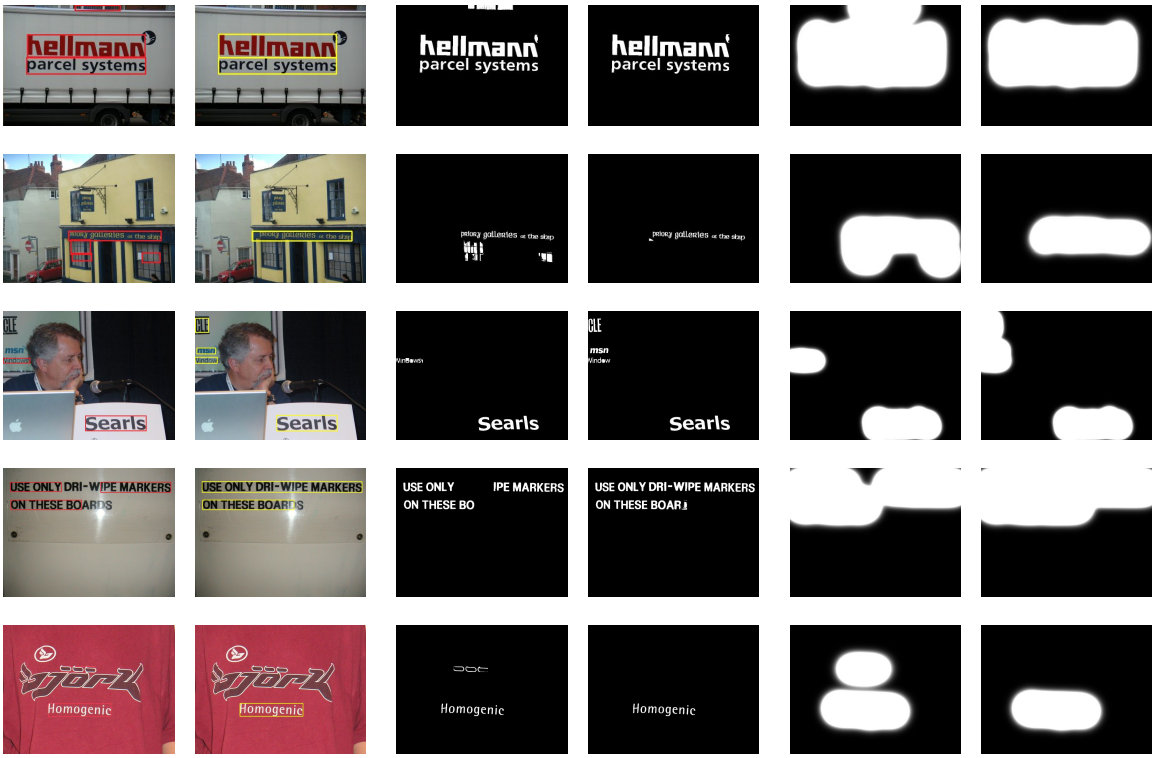


Figure 5.6: Sample images of the used text detections SWT (first row) and OCVTD (second row), the text segmentation approach for the SWT results (third row) and the OCVTD results (fourth row) and the associated text saliency images (fifth and sixth row).

suggested, we did not use any pixels on the boundary between p and q or within 10 pixels of the image's border. Figure 5.7 shows a sample image of the human fixation ground truth and the selected positive (as green dots) and negative (as red dots) samples using the above described procedure. The model's training set X was thus defined as the 10 selected positive and the 10 selected negative pixels for each image and each feature of the previous described feature matrix, yielding in $(n * 20)xN$ sampled feature matrix. Furthermore, a label vector Y is defined containing a label 1 for each positive sample and a label 0 for each negative sample of this matrix.

Some of the image features above described were computed in different data ranges since some of them are purely binary images, whereas others are continuous grayscale maps or probability distributions. To enable a useful classification between them, we needed to whiten X to acquire zero mean and unit variance, as proposed by Judd et al. The resulting whitening parameters for each feature column of X were then stored and reused for the testing phase.

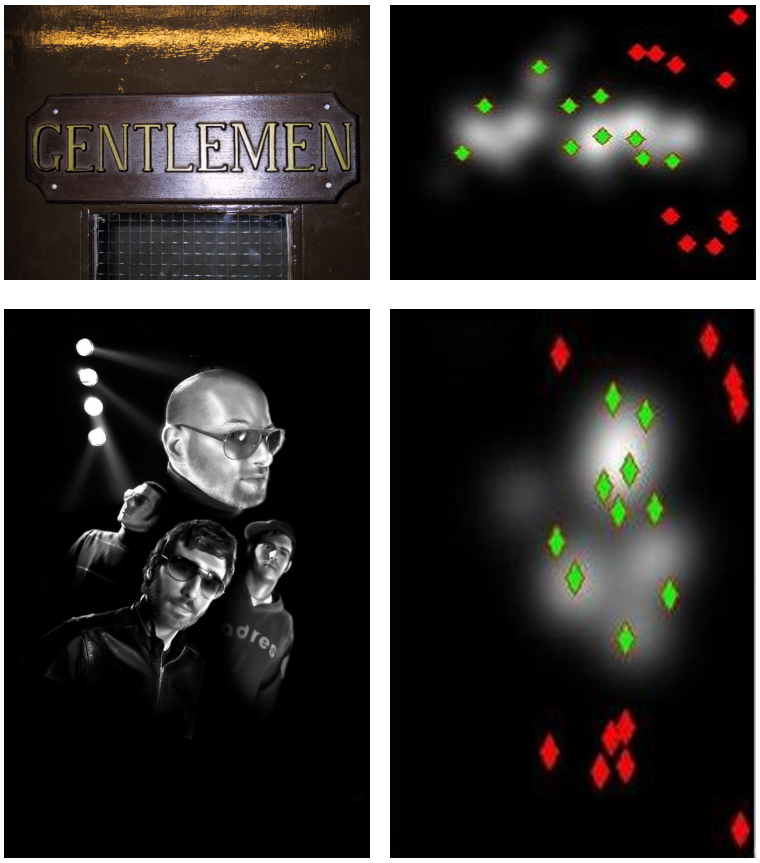


Figure 5.7: A sample selection of 10 positive (green dots) and 10 negative (red dots) sample pixels that were randomly chosen from the top 20% salient locations and the bottom 70% salient locations of the human fixation ground truth image.

Once the labeled feature matrix X was computed, a **classification procedure** was applied to achieve a weighting value for each feature, as described accordingly:

The classifier

The model was trained with a *Liblinear*[10] [Fan2008] support vector machine (SVM) with linear kernels. A linear SVM [CortesVapnik1995] is a supervised machine learning approach that is used for classification tasks and regression analysis. It produces a set of hyperplanes within a high-dimensional space filled with training data, where usually a good classification is acquired with the *maximum-margin* hyperplane that is defined as the maximum distance hyperplane to the closest data-points of both classes. The generalization error, a quality measure for classifications, usually decreases the larger the margin gets. Boser et al. [Boser1992] proposed an approach to train non-linear SVM classifiers by using Aizerman et al.'s [Aizerman1964] kernel trick. Applied on the SVM algorithm, it was used to fit the maximum-margin hyper-

[10]http://www.csie.ntu.edu.tw/~cjlin/liblinear/

plane, using a nonlinear transformation in a high dimensional transformed feature-space and therefore produce a nonlinear hyperplane in the original classifier space. Depending on the task a SVM can thus be trained not only with linear kernels, but for example with radial kernels or multiple kernel learning as proposed by Sonnenburg et al. [Sonnenburg2006].
Judd et al. [Judd2009] states that experimentations had shown that models with linear kernels performed as well as models trained with multiple-kernel for the given task and are additionally easier to interpret and faster to compute. Therefore we decided to stay with a linear kernel SVM to classify our attention model.

The training process

To train the model we used 1300 training images randomly chosen from our combined dataset yielding 13000 positive and 13000 negative samples for each feature. Hence the sampled feature matrix X had a dimension of $26000x34$ and the label vector Y a dimension of $26000x1$. We used the Matlab implementation of the liblinear SVM to train the our model with the following parameters:

- The bias b was activated by setting it to 1, to receive a bias term for the weighting vector. The online example of Judd et al.'s framework did not include a bias value, however, they state that they used the bias for the testing phase and moreover it makes no sense to deactivate it for this classification task.
- The so-called misclassification cost c was set to 1, since Judd et al. [Judd2009] already did a cross validation on different c values and stated that the performance between 1 and 10000 did not change, but rather got worse when c was smaller than 1.

The liblinear SVM train function needed X to be a sparse matrix for which we used the basic Matlab function. Then we used the whitening function, as described above, to create zero means and unit variance and started the training. The resulting model consisted of a weighting vector w that provided a weight value for each feature and a bias value b, as well as the weighting values described above that we need for normalizing in the testing phase. Figure 5.8 shows the resulting mean weights of the model trained with Judd et al.'s approach, our approach using the SWT text saliency and our approach using the OCVTD approach, as well as their standard deviation for 5 training trails. What we learned from this figure first of all is that the distance to center feature has a very strong negative value in all of the models, which backs up the center bias theory stated in chapter 2.1.2. Since this feature is saliency map that has it's most salient pixels at the border of an image and it's least salient pixels in the center (see figure 5.2) a strong negation of this feature will vote a stronger saliency the closer a pixel is located to the center. This of course has to do with the image database we used in our eye tracking study, where most of the supposedly interesting objects were located at the center of an image, and due to the fact that people tend to look for these objects near the center. Therefore, the center bias is that strongly mapped because the training of the model's weights is dependent on

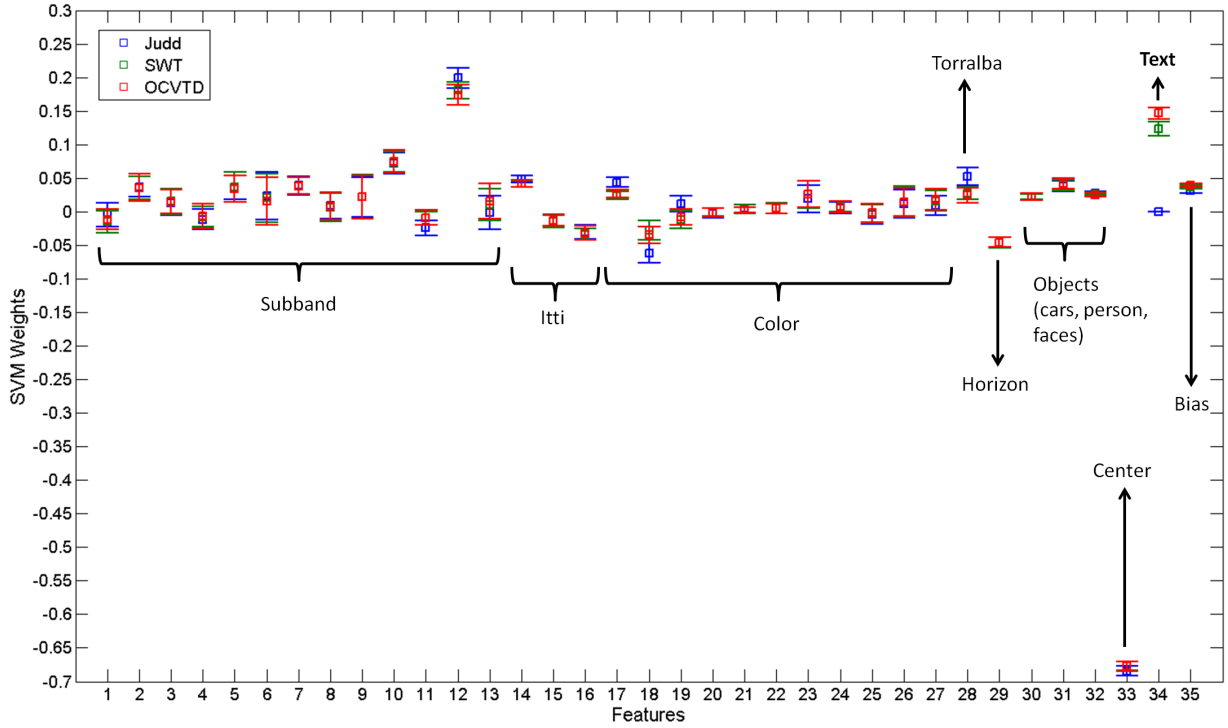


Figure 5.8: The mean trained model weighting values for each feature and the corresponding standard deviation for the retrained Judd et al. approach (blue), our approach using the SWT text saliency maps (green) and our approach using the OCVTD saliency maps (red) for 5 training trails. The text feature (number 34) was set to zero in this diagram for the original Judd et al. approach but is in fact not existing.

the human fixations we gathered in the eye tracking study. Furthermore the text feature's weight is slightly more important in the model trained with the OCVTD feature, which would encourage the hypothesis that a model's performance will also be higher when using a better text detector in the feature collection phase. We did a performance comparison of different models trained on the feature groups described above in chapter 6.

Saliency prediction

We will now explain how a prediction saliency map was computed using the model described above, all details on testing the model and it's performance will follow in chapter 6. When using a saliency model in applications (see section 2.3) the typical desired output of the model is thresholded image locations of the top i percent salient locations. In this project, as proposed by Judd et al. [Judd2009], the weights learned by the SVM classification are used to create a continuous saliency map for a given image. We use the same feature collection process as for the training to gather the input image's feature matrix x and then use the value of $xw + b$, where w and b

are the learned parameters of our text saliency model, to acquire a saliency value for each pixel. As stated above, we used the same normalization parameters to normalize the prediction image's feature matrix as in the training whithening process. Figure 5.9 shows a sample of a predicted saliency map using our model compared to the saliency map predicted with Judd et al.'s original model. As can be seen our model

(a) (b) (c)

Figure 5.9: A saliency map prediction with our model (c) compared to a predicted saliency map with the approach by Judd et al. [Judd2009] (b) of a sample image (a).

emphasized text region as strongly salient.

Our computational attention model approach in this thesis used the framework proposed by Judd et al. [Judd2009] and extended it with a bottom-up text saliency feature. With the newly collected human fixations of our eye tracking study (see chapter 4), on the combined natural scene images, as well as text focus images in natural scenes, we also added top-down text context to the model. We further tried to scale the weight of the text feature, as proposed by [Wolfe1989], to add yet another structural top-down improvement to the model. However, tests with several scaling factors showed that, even tough improvements occurred for some testing images, the mean prediction performance did not improve. We carried out several experiments to test our computational attention model's performance to predict human fixations and summarized them and their results in chapter 6. All in all the framework based on Judd et al.'s [Judd2009] framework provides a robust way to learn new computational attention models and the resulting weights vector enables an efficient computation of saliency maps based on those models.

Chapter 6

Experiments and Results

In this chapter we are describing how we carried out the experiments to test the computational attention model's performance, show their results and discuss them. As described in the previous chapter we trained our new model using 1300 randomly chosen training images from our image database. The continuous human fixation saliency maps we created as described in chapter 4.4, were used for the positive and negative sample selection we needed to train the SVM. As the ground truth for testing the performance of our model we did not use these continuous human fixation saliency maps but the actual human fixation points, as proposed by Judd et al. [Judd2009]. We used 190 testing images that were not used to train the model from our database and gathered the associated human fixation points for each of these testing images. Since our model is based on the approach of Judd et al. we wanted to compare our model to the original one, but we also retrained Judd et al.'s approach on our dataset (using our images and collected fixation maps) to compare the original and retrained approach to our model. We trained separate models using the text saliency features computed with the SWT approach as well as the OCVTD approach. To show how the other feature groups affected the model we also trained separate models for all used feature groups as well as a model using only the text saliency feature.
The performance of a model was tested pixel-wise, as proposed by Judd et al., by predicting a saliency value for each pixel of a test image. A predicted saliency map of the test image was created as described in chapter 5.1.2, by using the same feature collection process as for the training to gather the input image's feature matrix x and then used the value of $xw + b$, where w and b are trained parameters of our learned model, to acquire a saliency value for each pixel. Since these saliency maps are commonly thresholded when used in applications, we used the defined thresholds by Judd et al. at $n = \{1, 3, 5, 10, 15, 20, 25, 30\}$ percent saliency. We used the thresholded continuous human fixation saliency maps as the human fixation performance measure to compare with the others. We did 5 training trails for each separately trained model, randomly chose 1300 training images and used 190 test images which were not included in the associated training trail. We predicted a saliency map for each testing image of each feature model and thresholded them for each n percent saliency. Figure 6.1 shows a sample of the thresholded saliency maps at $n = 10$, 20 and 30 percent saliency for the top performing models Judd et al.'s original approach, the retrained version of it on our dataset, and our models including the two text saliency

approaches. For each n we computed the true positive rate of predicted salient pixels and

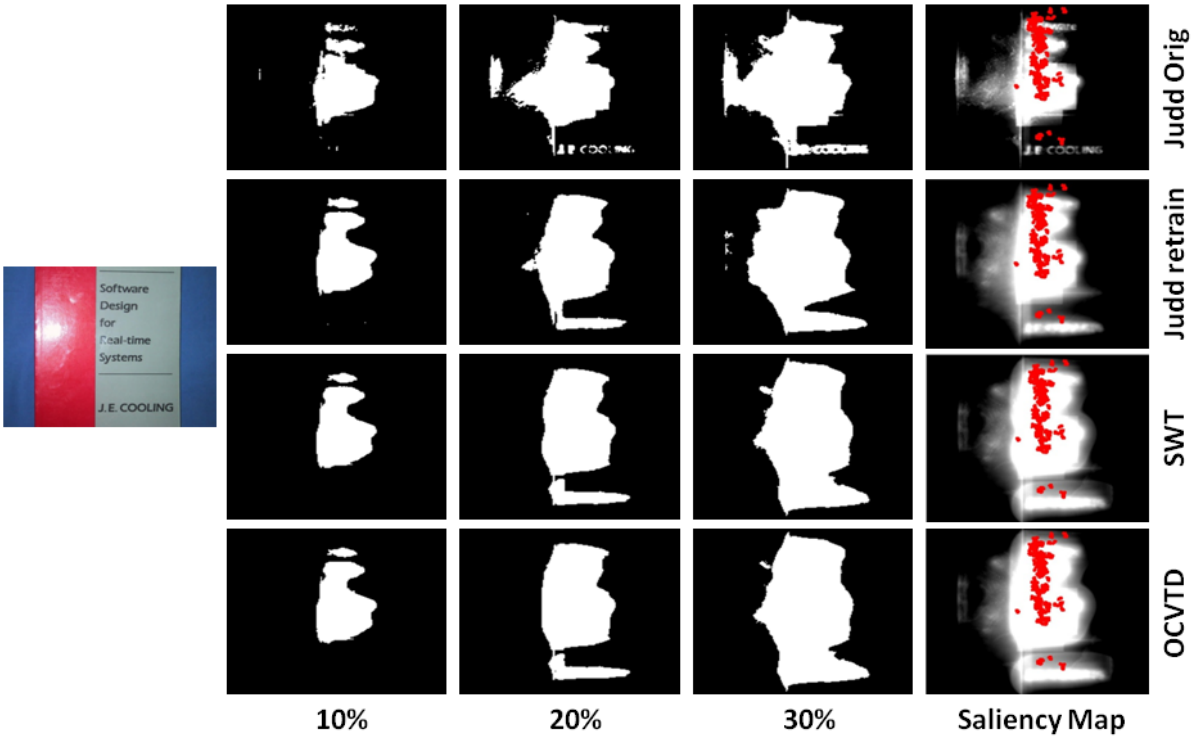


Figure 6.1: The thresholded predicted saliency maps at $n = 10$, 20 and 30 percent saliency for the sample image on the left, predicted with Judd et al.'s original model, a retrained version of it and our models using the SWT and OCVTD text saliency approaches. We also superimposed the ground truth human fixation points as red dots on the predicted continuous saliency maps without an applied threshold.

ground truth fixation point pixels. Figure 6.2 shows a receiver operating characteristic (ROC) curve of the mean true positive rate performance over all trails for each n percent salient predictions. The models trained with all features including a text feature as well as the retrained model based on Judd et al. outperformed the other models, including the original Judd et al. model. The model using the better text detection algorithm OCVTD performed slightly better than the one using the SWT approach. Also both models trained with text saliency performed slightly better than the retrained model of Judd et al., which was after all trained on a database containing a lot of text context. Figure 6.3 shows a boxplot of the four leading models for the 30 percent threshold maps in detail, where the red line represents the mean value and the blue box the standard deviation of each models performance results. As expected the model trained only on the center feature performed better than the other feature sets without it, followed by the combined feature sets of all other features but the center. Other than in the published results of Judd et al. [Judd2009] did the model trained using the subband features [SimoncelliFreeman1995] perform slightly

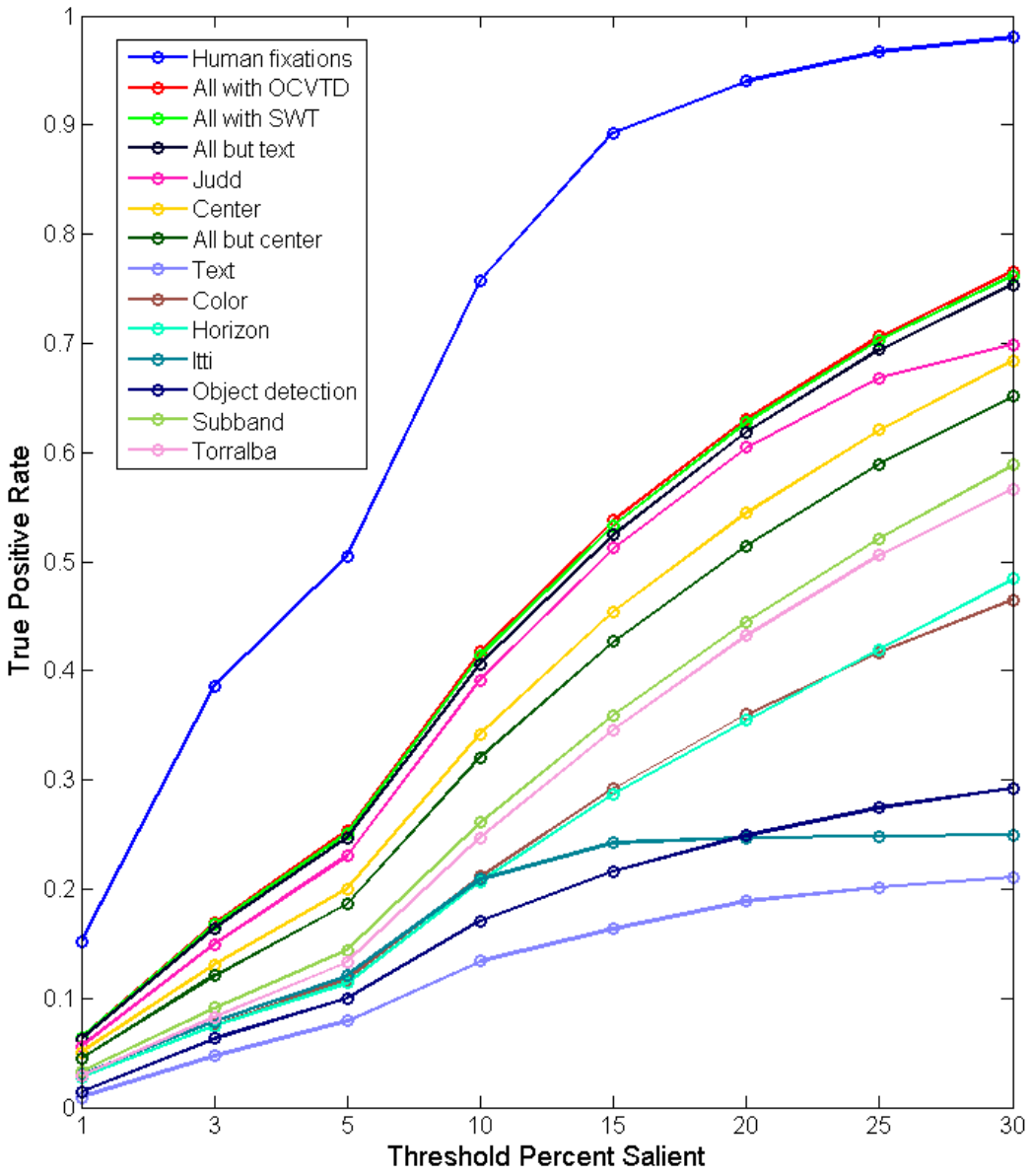


Figure 6.2: The ROC curve of performance for models trained on our dataset using different feature sets. The model's performance was tested on several thresholds which indicate the percentage of saliency that we used to threshold the predicted saliency maps.

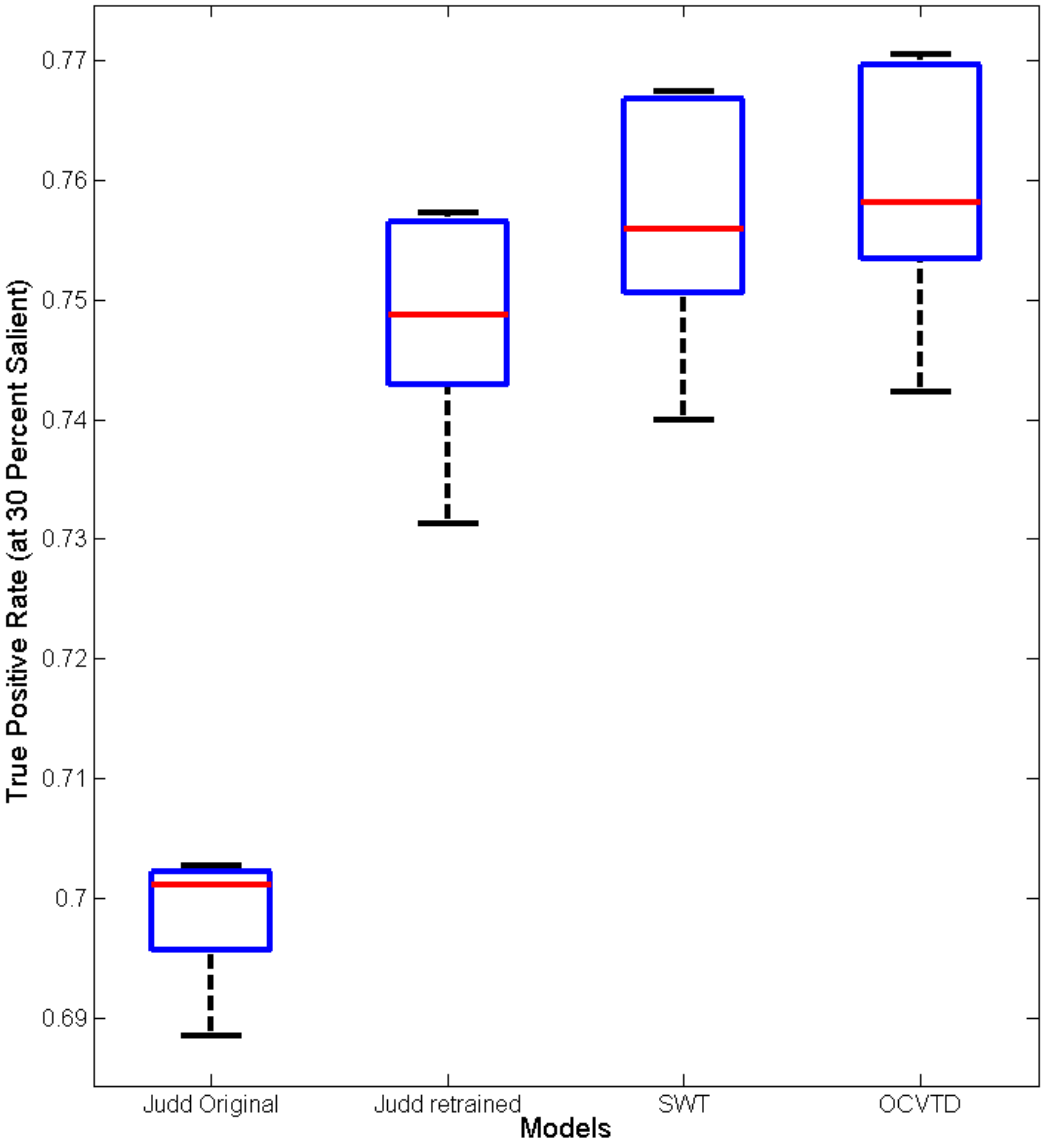


Figure 6.3: The mean values and standard deviations of the area under the ROC performances, thresholded at 30% saliency, for the original Judd et al. model, a retrained version on our dataset and our approach using both SWT and OCVTD text saliency approaches.

better than the model trained with the so-called Torralba feature map (based on the approach of [Rosenholtz1999] and [OliviaTorralba2001]). The models trained with color and horizon features performed close to 50% at 30% saliency which is lower than in Judd et al.' published results. These differences occurred most likely because we used 35% more text images in our database as well as fixation data from 15 different participants. The visual properties of test images may also be the reason why the Itti feature (based on the STB [WaltherKoch2006]) reaches an unexpectedly low performance of only 25% at 30% saliency. Object detection features alone are known to be no sufficient predictors for natural scene images [Judd2009], they lack the potential for reasonable predictions when no objects are present. The same applies for the text saliency feature, which on its own yielded even lower results than the object detection, which however contained three different object detectors (face, car and person), and therefore was expected to perform better.

To show the impact of the center bias we translated all of our stimuli images and the associated human fixation positions to random positions in an accordingly larger image. By performing this image transformation we moved the interesting objects away from the center which should neutralize the distance to center features importance and represent how important the contributions of other features are. Figure 6.4 shows the resulting weights of our text saliency model trained on these translated stimuli images. If you compare them

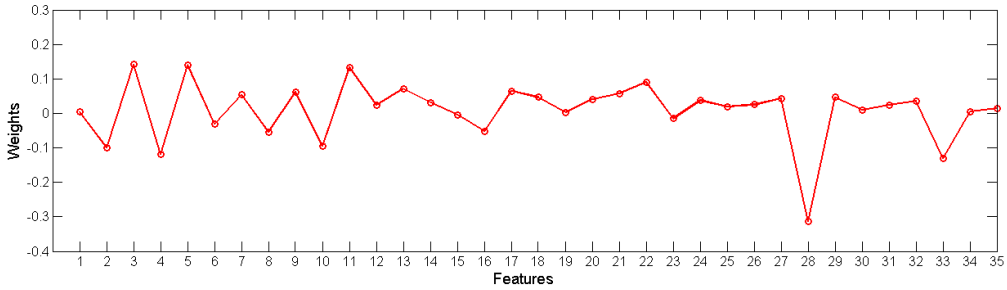


Figure 6.4: The SVM weights learned from images of our database translated to random positions. As can be seen, distance to center feature weight (number 33) is no longer the most important.

to the mean weight values in figure 5.8 it is obvious that the distance to center feature (number 33) has lost the most importance and the weights overall are far less diverse with a maximum difference of 0.455. The most important feature, and therefore most beneficial after the distance to the center, seems to be the so-called Torralba feature based on the approach of [Rosenholtz1999] and [OliviaTorralba2001], which also concurs with the findings Judd et al. [Judd2009] provided on their dataset. After that the subband features score the most important weights, which consequently makes sense since the Torralba feature is based on subband features. Text and object features seem to provide less benefit then most of the other features, however are still useful in combination with other features as we can see in the computed results of figure 6.2. Figure 6.5 shows some example evaluations

we performed using the translated model, as expected the results resemble saliency maps computed with Olivia and Torralba's approach [OliviaTorralba2001].

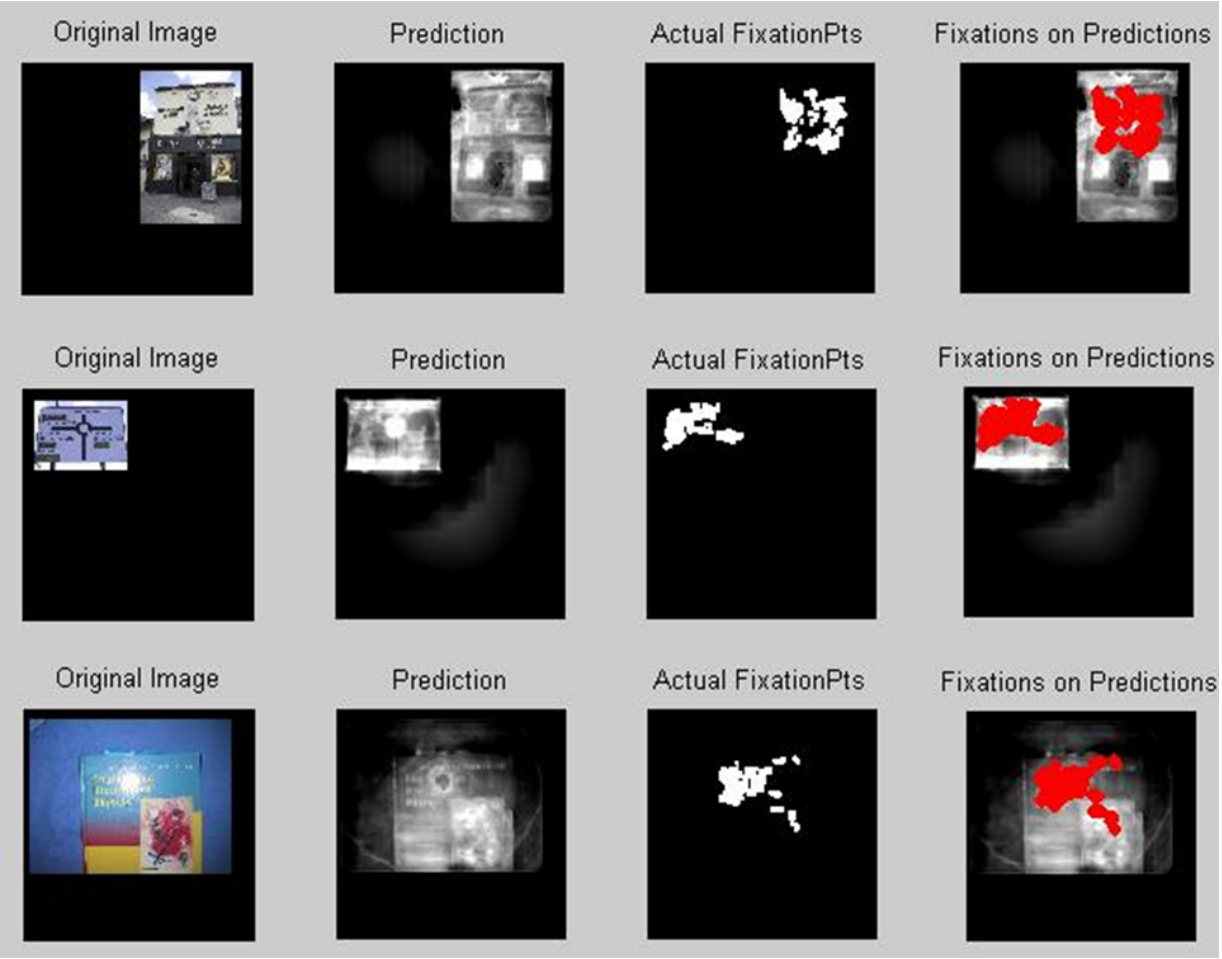


Figure 6.5: Some sample evaluations on the translated images with the model weights shown in figure 6.4.

To compare our model's performance to other computational saliency models we used the saliency benchmark[1] proposed by [Judd2012]. This benchmark system was introduced to provide a way for researchers to compare their computational attention models with others. The used metrics to compare the models' performances are defined as:

The area under the ROC curve (AUR) [Green1966] which measures the correlation of saliency maps and fixation points, hence the probability that a fixation point is voted correctly from the saliency map.

[1] http://people.csail.mit.edu/tjudd/SaliencyBenchmark/index.html

The Earth mover's distance (EMD) [Rubner2000] which measures the distance of two probability distributions within an area, where the EMD indicates the overall difference of these distributions. A smaller EMD equals a smaller distance and hence more similarity of the two areas.

The similarity measure indicates how similar two distributions are [Judd2012], and computes their overlap. The sum of the distributions is normalized to equal one and the similarity is computed as the sum of the minimum value of each value of the overlapping distributions. Therefore, a score of 1 means the areas are completely overlapping while 0 means they share no similar value.

To compute the scores for these three metrics and compare them to other models one has to download 300 scene images from the benchmarks webpage and compute their saliency maps. We computed the saliency map using the model including the SWT text saliency feature and uploaded them. The results are listed in table 6.1 where we ranked as fourth from a total of 22 competing approaches (the full list can be seen on the above mentioned webpage).

Based on the AUR metric we further compared the mean AUR results for our 5 test trails (again using 190 test images) without thresholding the predicted saliency maps. Figure 6.6 shows some sample results of predicted continuous saliency maps for the top four performing models as well as a superimposed human fixation ground truth. As can be seen the predicted saliency maps are fairly similar since they are all based on Judd et al.'s approach. However differences, for example regarding the text features, can be seen in image three where detected text regions are significantly brighter when using the text saliency approaches. As shown in the boxplot in figure 6.7 there are only slight differences between the Judd et al. model and its retrained version, and our text saliency approaches. However the state-of-the-art models ranked in the benchmark system above, including our model, also show only slight differences in the score metrics.

The fact that we performed better than Judd et al. on our dataset and worse on their benchmark system is again deducible by the missing text context in the benchmark database images. Our approach only detected 12 images containing text areas within the 300 benchmark images, which is only 4%. Furthermore, since our dataset contains 38% scene text images and participants obviously looked actively at these areas, our model is also trained to weighting text like areas higher then a model trained on fixation data with less text context.

Since text saliency is still largely omitted in state-of-the-art computational attention model, we were not able to find other published datasets with more focus on text context. Overall our model performed best on the proposed dataset, however not significantly better than the retrained model of Judd et al. on the data containing text context.

Model	AUR (higher is better)	Similarity (higher is better)	EMD (lower is better)
Humans*	0.922	1	0
Bio-inspired hierarchical features (not published)	0.8192	0.5123	3.0129
Judd et al. [Judd2009]	0.811	0.506	3.13
CovSal [Erdem2013]	0.8056	0.5018	3.1092
Our model	**0.8095**	**0.5008**	**3.2303**
Tavakoli et al. [Tavakoli2011]	0.8033	0.4952	3.3488
Region Contrast [Cheng2011]	0.7922	0.4705	3.4180
Multi-Resolution AIM [Advani2013]	0.7719	0.4711	3.3635
Center*	0.783	0.451	3.719
Random Center Surround Saliency [Vikram2012]	0.7719	0.4711	3.3635

Table 6.1: A saliency benchmark comparison based on the benchmark system of [Judd2012]. Computational attention models and *baseline models are compared by the scores of three different metrics.

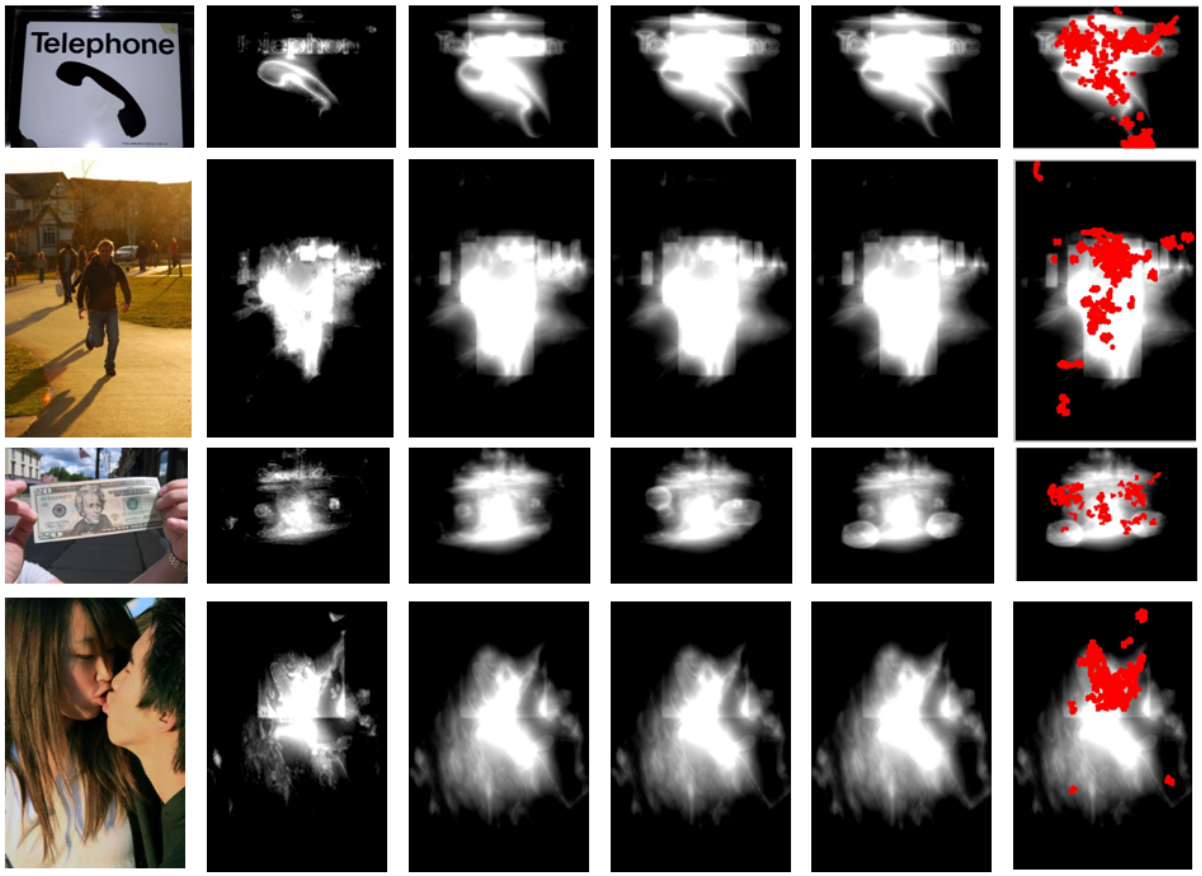


Figure 6.6: Sample predicted continuous saliency maps for the four top performing models according to the AUR scores. The saliency maps for each image are arranged as the following: the second column saliency maps are computed using Judd et al.'s model, the third column maps are predicted using Judd et al. approach retrained on our text context database and the fourth and fifth column indicate the predictions using the SWT based and OCVTD based text saliency model. The last column shows the OCVTD predictions again with the superimposed human fixation ground truth points.

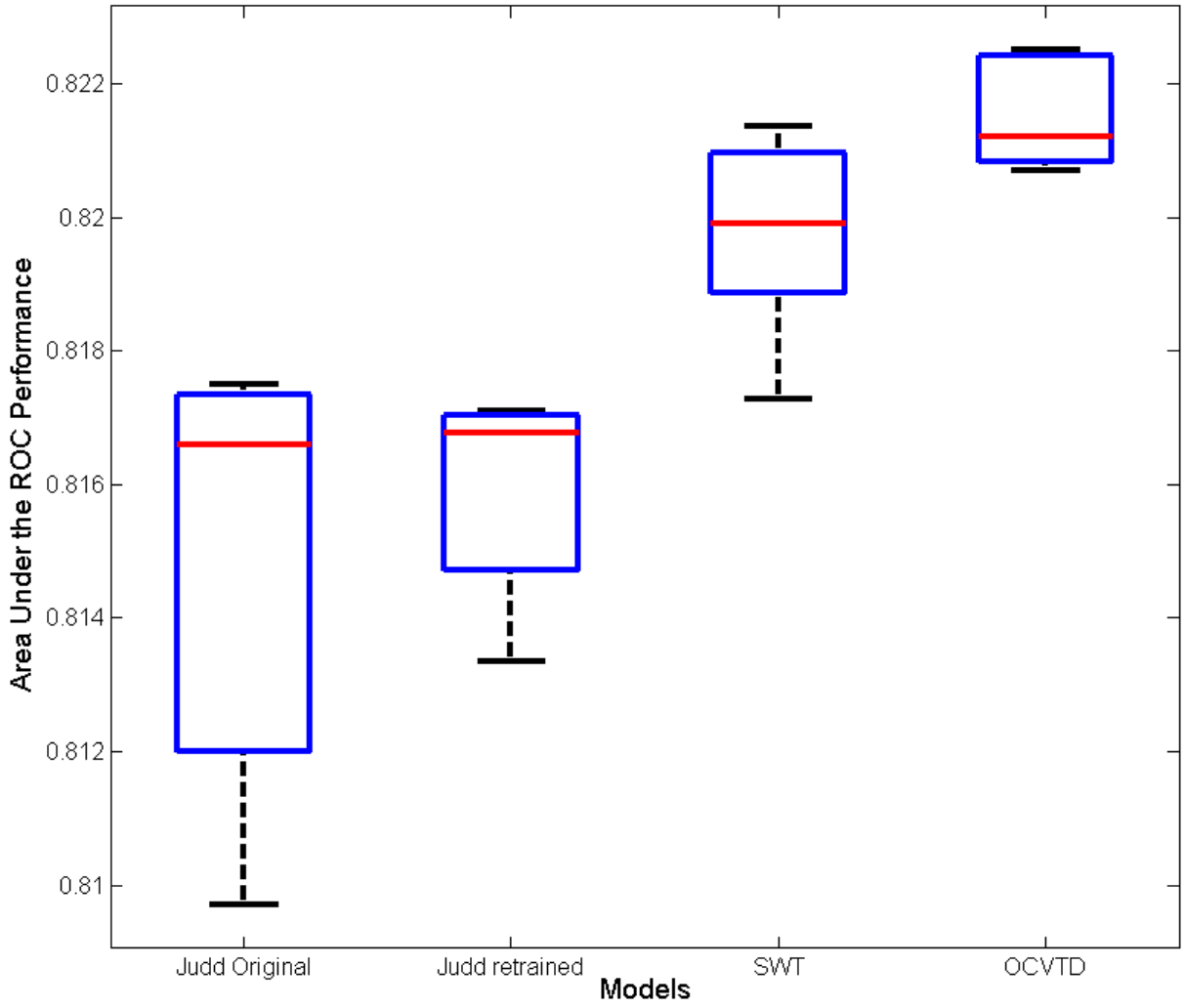


Figure 6.7: The mean AUR performance on the non-thresholded predicted saliency maps computed by the four model: Judd et al. original model, Judd et al.'s approach retrained on our dataset, our approach using the SWT text saliency features and our approach using the OCVTD saliency features.

Chapter 7

Conclusion

In this thesis we engaged the largely omitted topic of computational attention models focusing on text saliency. To use both natural scene images as well as scene text images for large scale attention studies, we combined the largest available natural scene database used for attention studies with a well known scene text database. We carried out an eye tracking user study using this database and gathered eye movement data from 15 participants. We learned a new computational model of attention on the hypothesis that trained text context from our database as well as a newly added text saliency feature will improve a current state-of-the-art attention model. To our knowledge, our combined database of 1498 stimuli images is the largest natural scene database containing data collected from an eye tracking study. We introduce a novel approach to compute text saliency maps, which performs slightly better than using a naive binarization of detected text regions. The performance results on our database have shown that we outperformed the state-of-the-art computational attention model without text saliency our thesis was based on. However, when retrained with our database, the original approach also learns the text context from human fixations on text images and the improvement of the text saliency feature alone is not as significant as expected. We have shown that the use of different text detectors for our text saliency approach yields only slight differences, but a better performing text detector also achieves better performing saliency models. On a benchmark system for general computational attention models, our model scored the fourth rank out of 22 models. We were ranked behind the original state-of-the-art model we based our approach on, and conclude that this is due to the lack of sufficient scene text images in the benchmark image database. However, we outperformed several general state-of-the-art models of recent years and only achieved about one percent less true detection rate than the first ranked model.

We conclude that using a text saliency feature provides an improvement to computational attention models that are used in fields where text context is important. The performance improvement of the model is depending on the used text detector and therefore the text saliency feature's performance. We further conclude that including top-down text context information by training the model using a fair amount of scene text images yields a significant improvement on databases with scene text images. Therefore, computational attention models using features to handle the center bias and features that emphasize

text properties, trained on a database containing text context, perform almost as good as models that include text saliency features as well. Simply scaling a text features weight after the model is learned seems to have no positive effect on the model's performance.

All in all the model's performance is promising for applications that use computational attention models in natural scenes and are focusing on the role of scene text. Since this thesis was carried out in cooperation with a company using this systems, we are looking forward on using our model in the field and testing its predictability of human fixations in real world scenarios.

7.1 Future work

For future work in this field of research we would be interested to see how even more recent text detector algorithms would change the model's performance. Furthermore we would be interested how the model would perform when trained on different datasets containing even more scene text focus than our database provided. And of course testing other top-down influences like additional background knowledge on text context or other machine learning approaches would be needed to further investigate the influence of text context in human visual attention.

As stated above, the computational attention model for this thesis was carried out in cooperation with a company in order to acquire a scene text focused human fixation predictor. There are some projects planned in the near future in which we will be able to test the model's performance in predicting human fixations based on a real world task like text guided navigation. A task like navigation is performed with a moving subject and therefore does not exactly apply on a model trained with a static eye tracker and random scene images. Therefore, we would further be interested in a model that is learned based on a scene video and associated eye movement data recorded with a mobile eye tracking device. However, since there is not yet a clear definition that we know of on how to compute real fixations with mobile eye trackers, a meaningful approach on this matter is not yet possible.

List of Figures

List of Tables

Bibliography

[Achanta2008] Radhakrishna Achanta, Francisco Estrada, Patricia Wils, and Sabine Susstrunk. *Salient region detection and segmentation. Computer Vision Systems*, volume 5008:pp. 66–75, **2008**.

[Advani2013] Siddharth Advani, John Sustersic, Kevin Irick, and Vijaykrishnan Narayanan. *A multi-resolution saliency framework to drive foveation.* In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pp. 2596–2600. IEEE, **2013**.

[Aizerman1964] A Aizerman, Emmanuel M Braverman, and LI Rozoner. *Theoretical foundations of the potential function method in pattern recognition learning. Automation and remote control*, volume 25:pp. 821–837, **1964**.

[Boser1992] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. *A training algorithm for optimal margin classifiers.* In *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 144–152. ACM, **1992**.

[BruceTsotsos2009] N D B Bruce and J K Tsotsos. *Saliency, attention, and visual search: An information theoretic approach. Journal of Vision*, volume 9(3):pp. 1–24, **2009**.

[Butko2008] N.J. Butko, Lingyun Zhang Lingyun Zhang, G.W. Cottrell, and J.R. Movellan. *Visual saliency model for robot cameras. 2008 IEEE International Conference on Robotics and Automation*, **2008**.

[Canny1986] J Canny. *A computational approach to edge detection. IEEE transactions on pattern analysis and machine intelligence*, volume 8(6):pp. 679–698, **1986**.

[Cerf2009] Moran Cerf, E Paxon Frady, and Christof Koch. *Faces and text attract gaze independent of the task: Experimental data and computer model. Journal of Vision*, volume 9(12):pp. 10.1–15, **2009**.

[ChenYuille2004] Xiangrong Chen Xiangrong Chen and A.L. Yuille. *Detecting and reading text in natural scenes*, **2004**.

[Cheng2011] Ming-Ming Cheng, Guo-Xin Zhang, Niloy J. Mitra, Xiaolei

Huang, and Shi-Min Hu. *Global contrast based salient region detection*. In *IEEE CVPR*, pp. 409–416. **2011**.

[Clark1988] J. J. Clark and N. J. Ferrier. *Modal control of an attentive vision system*. In *Proc. 2nd Int. Conf. on Computer Vision*, pp. 514–522. Tarpon Springs, USA, **December 1988**.

[ClarkFerrier1989] J.J. Clark and N.J. Ferrier. *Control of visual attention in mobile robots*, **1989**.

[Corbetta1998] Maurizio Corbetta. *Frontoparietal cortical networks for directing attention and the eye to visual locations: identical, independent, or overlapping neural systems? Proceedings of the National Academy of Sciences*, volume 95(3):pp. 831–838, **1998**.

[CorbettaShulman2002] Maurizio Corbetta and Gordon L Shulman. *Control of goal-directed and stimulus-driven attention in the brain. Nature reviews. Neuroscience*, volume 3(3):pp. 201–215, **2002**.

[CortesVapnik1995] Corinna Cortes and Vladimir Vapnik. *Support-vector networks. Machine Learning*, volume 20(3):pp. 273–297, **1995**. ISSN 0885-6125.

[DesimoneDuncan1995] R Desimone and J Duncan. *Neural mechanisms of selective visual attention. Annual Reviews in Neuroscience*, volume 18:pp. 193–222, **1995**.

[DodgeCline1901] Raymond Dodge and Thomas Sparks Cline. *The angle velocity of eye movements. Psychological Review*, volume 8(2):p. 145, **1901**.

[DraperLionelle2005] Bruce A. Draper and Albert Lionelle. *Evaluation of selective attention under similarity transformations. Computer Vision and Image Understanding*, volume 100(1-2):pp. 152–171, **2005**.

[ElazaryItti2010] Lior Elazary and Laurent Itti. *A bayesian model for efficient visual search and recognition. Vision research*, volume 50(14):pp. 1338–1352, **2010**.

[Epshtein2010] Boris Epshtein. *Detecting text in natural scenes with stroke width transform. Image Rochester NY*, volume 93(d):pp. 1–8, **2010**.

[Erdem2013] Erkut Erdem and Aykut Erdem. *Visual saliency estimation by nonlinearly integrating features using region covariances. Journal of vision*, volume 13(4):p. 11, **2013**.

[Fan2008] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. *Liblinear: A library for large linear classification. J. Mach. Learn. Res.*, volume 9:pp. 1871–1874, **June 2008**. ISSN 1532-4435.

[Felzenszwalb2008] Pedro Felzenszwalb, David McAllester, and Deva Ramanan. *A discriminatively trained, multiscale, deformable part model.* In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pp. 1–8. IEEE, **2008**.

[Vasconcelos2009] Dashan Gao and Nuno Vasconcelos. *Decision-theoretic saliency: Computational principles, biological plausibility, and implications for neurophysiology and psychophysics. Neural Comput.*, volume 21(1):pp. 239–271, **January 2009**. ISSN 0899-7667.

[Gllavata2004] J. Gllavata, R. Ewerth, and B. Freisleben. *Text detection in images based on unsupervised classification of high-frequency wavelet coefficients*, **2004**.

[Goferman2010] S. Goferman, L. Zelnik-Manor, and A. Tal. *Context-aware saliency detection*, **2010**.

[GomezKaratzas2013] Lluis Gomez and Dimosthenis Karatzas. *Multi-script text extraction from natural scenes.* In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pp. 467–471. IEEE, **2013**.

[Green1966] David Marvin Green, John A Swets, et al. *Signal detection theory and psychophysics*, volume 1974. Wiley New York, **1966**.

[Harel2007] Jonathan Harel, Christof Koch, and Pietro Perona. *Graph-based visual saliency. America*, volume 19(2):p. 545, **2007**.

[Henderson2005] John M Henderson, James R Brockmole, Monica S Castelhano, Michael Mack, Roger Van Gompel, Martin Fischer, Wayne Murray, Robin Hill, and Eds Eye. *visual saliency does not account for eye movements during visual search in real world scenes. Mind*, pp. 1–41, **2005**.

[Hidalgo2009] Barbara Hidalgo-Sotelo, Antonio Torralba, Krista A. Ehinger, and Aude Oliva. *Modelling search for people in 900 scenes: A combined source model of eye guidance. Visual Cognition*, volume 17(6-7):pp. 945–978, **2009**.

[Hoffman1998] James E Hoffman. *Visual attention and eye movements. Attention*, volume 31:pp. 119–153, **1998**.

[Holmqvist2011] Kenneth Holmqvist, Marcus Nyström, Richard Andersson, Richard Dewhurst, Halszka Jarodzka, and Joost Van de Weijer. *Eye tracking: A comprehensive guide to methods and measures.* Oxford University Press, **2011**.

[Horn1986] Berthold K P Horn. *Robot Vision*, volume 4931. The MIT Press, **1986**.

[IttiKoch2001] L. Itti and C. Koch. *Computational modelling of visual attention. Nature Reviews Neuroscience*, volume 2(3):pp. 194–203, **Mar 2001**.

[Itti1998] 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*, volume 20(11):pp. 1254–1259, **1998**. ISSN 01628828.

[IttiNavalpakkam2010] L. Itti and V. Navalpakkam. *Optimal feature gain modulation during visual search. Journal of Vision*, volume 6(6):pp. 454–454, **2010**.

[JainBhattacharjee1992] Anil K. Jain and Sushil Bhattacharjee. *Text segmentation using gabor filters for automatic document processing. Machine Vision and Applications*, volume 5(3):pp. 169–184, **1992**.

[James1890] William James. *The principles of psychology. Harvard UP, Cambridge, MA*, **1890**.

[Judd2011a] Tilke Judd. *Understanding and predicting where people look in images.* Ph.D. thesis, Massachusetts Institute of Technology, **2011**.

[Judd2011b] Tilke Judd, Fredo Durand, and Antonio Torralba. *Fixations on low-resolution images. Journal of vision*, volume 11(4):pp. 1–20, **2011**.

[Judd2012] Tilke Judd, Frédo Durand, and Antonio Torralba. *A benchmark of computational models of saliency to predict human fixations.* **2012**.

[Judd2009] Tilke Judd, Krista Ehinger, Frédo Durand, and Antonio Torralba. *Learning to predict where humans look.* In *IEEE International Conference on Computer Vision (ICCV)*. **2009**.

[Jung2004] Keechul Jung, Kwang In Kim, and Anil K. Jain. *Text information extraction in images and video: a survey. Pattern Recognition*, pp. 977–997, **2004**.

[JustCarpenter1980] Marcel Adam Just and Patricia A Carpenter. *A theory of reading: From eye fixations to comprehension. Psychological review*, volume 87:pp. 329–354, **1980**.

[Kasar2007] Thotreingam Kasar, Jayant Kumar, and AG Ramakrishnan. *Font and background color independent text binarization.* In *Second international workshop on camera-based document analysis and recognition*, pp. 3–9. **2007**.

[KochUllman1985] C. Koch and S. Ullman. *Shifts in selective visual attention: To-*

wards the underlying neural circuitry. Human Neurobiology, volume 4:pp. 219–227, **1985**.

[Konuskan2008] Fatma Konuskan. *Visual saliency and biological inspired text detection.* **2008**.

[Lee2011] Jung-Jin Lee, Pyoung-Hean Lee, Seong-Whan Lee, Alan Yuille, and Christof Koch. *Adaboost for text detection in natural scene*, **2011**.

[Li2000] H Li, D Doermann, and O Kia. *Automatic text detection and tracking in digital video. IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, volume 9(1):pp. 147–156, **2000**.

[Liang2005] Jian Liang, David Doermann, and Huiping Li. *Camera-based analysis of text and documents: a survey. International Journal of Document Analysis and Recognition (IJDAR)*, volume 7(2-3):pp. 84–104, **2005**. ISSN 1433-2833.

[Liu2005] Y. Liu, S. Goto, and T. Ikenaga. *A robust algorithm for text detection in color images*, **2005**.

[Vasconcelos2010] Vijay Mahadevan and Nuno Vasconcelos. *Spatiotemporal saliency in dynamic scenes. IEEE Trans. Pattern Anal. Mach. Intell.*, volume 32(1):pp. 171–177, **2010**.

[Mahadevan2013] Vijay Mahadevan and Nuno Vasconcelos. *Biologically inspired object tracking using center-surround saliency mechanisms. IEEE Trans. Pattern Anal. Mach. Intell.*, volume 35(3):pp. 541–554, **2013**.

[Mancas-Thillou2007] Céline Mancas-Thillou, Silvio Ferreira, Jonathan Demeyer, Christophe Minetti, and Bernard Gosselin. *A multifunctional reading assistant for the visually impaired. EURASIP Journal on Image and Video Processing*, volume 2007(1):p. 064295.

[Marchesotti2009] L. Marchesotti, C. Cifarelli, and G. Csurka. *A framework for visual saliency detection with applications to image thumbnailing*, **2009**.

[Matas2004] Jiri Matas, Ondrej Chum, Martin Urban, and Tomás Pajdla. *Robust wide-baseline stereo from maximally stable extremal regions. Image and vision computing*, volume 22(10):pp. 761–767, **2004**.

[Miau2001] Florence Miau. *Neuromorphic algorithms for computer vision and attention*, volume 4479, pp. 12–23. SPIE, **2001**.

[NavalpakkamItti2006] V Navalpakkam and Laurent Itti. *An Integrated Model of Top-*

down and Bottom-up Attention for Optimal Object Detection, pp. 2049–2056. **2006**.

[NeumannMatas2012] L. Neumann and J. Matas. *Real-time scene text localization and recognition.* In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 3538–3545. **2012**. ISSN 1063-6919.

[NeumannMatas2011] Lukas Neumann and Jiri Matas. *A method for text localization and recognition in real-world images.* In *Computer Vision–ACCV 2010*, pp. 770–783. Springer, **2011**.

[Nothdurft2000] H Nothdurft. *Salience from feature contrast: temporal properties of saliency mechanisms. Vision research*, volume 40(18):pp. 2421–2435, **2000**.

[Oh2013] Il-Seok Oh, Jinseon Lee, and Aditi Majumder. *Multi-scale image segmentation using mser.* In *Computer Analysis of Images and Patterns*, pp. 201–208. Springer, **2013**.

[OliviaTorralba2001] Aude Oliva and Antonio Torralba. *Modeling the shape of the scene: A holistic representation of the spatial envelope. International journal of computer vision*, volume 42(3):pp. 145–175, **2001**.

[ParkhurstNiebur2003] Derrick J Parkhurst and Ernst Niebur. *Scene content selected by active vision. Spatial vision*, volume 16(2):pp. 125–154, **2003**.

[PooleBall2006] A Poole and L J Ball. *Eye Tracking in Human-Computer Interaction and Usability Research: Current Status and Future Prospects.* Idea Group Inc, **2006**.

[Tavakoli2011] Hamed Rezazadegan Tavakoli, Esa Rahtu, and Janne Heikkilä. *Fast and efficient saliency detection using sparse sampling and kernel density estimation.* In Anders Heyden and Fredrik Kahl, editors, *Image Analysis*, volume 6688 of *Lecture Notes in Computer Science*, pp. 666–675. Springer Berlin Heidelberg, **2011**. ISBN 978-3-642-21226-0.

[Rosenholtz1999] Ruth Rosenholtz. *A simple saliency model predicts a number of motion popout phenomena. Vision research*, volume 39(19):pp. 3157–3163, **1999**.

[Rubner2000] Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. *The earth mover's distance as a metric for image retrieval. International Journal of Computer Vision*, volume 40(2):pp. 99–121, **2000**.

[Russell2008] Bryan C. Russell, Antonio Torralba, Kevin P. Murphy, and William T. Freeman. *Labelme: A database and web-based tool for image annotation. International Journal of Computer Vi-*

sion, volume 77(1-3):pp. 157–173, **2008**.

[Santella2006] Anthony Santella, Maneesh Agrawala, Doug Decarlo, David Salesin, and Michael Cohen. *Gaze-based interaction for semi-automatic photo cropping.* In *In CHI 2006*, pp. 771–780. ACM, **2006**.

[SantellaDeCarlo2002] Anthony Santella and Doug DeCarlo. *Stylization and abstraction of photographs. ACM Transactions on Graphics*, volume 21(3), **2002**.

[Shahab2012] Asif Shahab, Faisal Shafait, Andreas Dengel, and Seiichi Uchida. *How salient is scene text?* In *IAPR International Workshop on Document Analysis Systems. IAPR International Workshop on Document Analysis Systems (DAS-12), 10th, March 27-29, Gold Coast„ Queensland, Australia.* IEEE, **3 2012**.

[SiagianItti2009] Christian Siagian and Laurent Itti. *Biologically inspired mobile robot vision localization. IEEE Transactions on Robotics*, volume 25(4):pp. 1–13, **2009**.

[SimoncelliFreeman1995] E.P. Simoncelli and W.T. Freeman. *The steerable pyramid: a flexible architecture for multi-scale derivative computation*, **1995**.

[Sonnenburg2006] Sören Sonnenburg, Gunnar Rätsch, Christin Schäfer, and Bernhard Schölkopf. *Large scale multiple kernel learning. The Journal of Machine Learning Research*, volume 7:pp. 1531–1565, **2006**.

[Lucas2003] Lucas Panaretos Sosa, S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, and R. Young. *Icdar 2003 robust reading competitions.* In *In Proceedings of the Seventh International Conference on Document Analysis and Recognition*, pp. 682–687. IEEE Press, **2003**.

[Subramanian2007] K. Subramanian, P. Natarajan, M. Decerbo, and D. Castanon. *Character-stroke detection for text-localization and extraction*, **2007**.

[Thillou2005] Céline Thillou, Silvio Ferreira, and Bernard Gosselin. *An embedded application for degraded text recognition. EURASIP Journal on applied signal processing*, volume 2005:pp. 2127–2135, **2005**.

[Torralba2006] Antonio Torralba, Aude Oliva, Monica S Castelhano, and John M Henderson. *Contextual guidance of eye movements and attention in real-world scenes: the role of global features in object search. Psychological review*, volume 113(4):pp. 766–786, **2006**.

[TorralbaSinha2001] Antonio Torralba and Pawan Sinha. *Statistical context priming for object detection.* In *Computer Vision, 2001. ICCV 2001. Pro-*

ceedings. Eighth IEEE International Conference on, volume 1, pp. 763–770. IEEE, **2001**.

[TreismanGelade1980] A M Treisman and G Gelade. *A feature-integration theory of attention. Cognit Psychol*, volume 12(1):pp. 97–136, **January 1980**.

[Tseng2009] Po-He Tseng, Ran Carmi, Ian G M Cameron, Douglas P Munoz, and Laurent Itti. *Quantifying center bias of observers in free viewing of dynamic natural scenes. Journal of vision*, volume 9(7):p. 4, **2009**.

[Valenti2009] Roberto Valenti, Nicu Sebe, and Theo Gevers. *Image saliency by isocentric curvedness and color*, **2009**.

[Varshney2005] Amitabh Varshney, David W. Jacobs, and Chang Ha Lee. *Mesh saliency. ACM Transactions on Graphics*, volume 24(3):p. 659, **2005**.

[Vikram2012] Tadmeri Narayan Vikram, Marko Tscherepanow, and Britta Wrede. *A saliency map based on sampling an image into random rectangular regions of interest. Pattern Recognition*, volume 45(9):pp. 3114 – 3124, **2012**. ISSN 0031-3203. Best Papers of Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA'2011).

[VincentTatler2009] Benjamin T. Vincent and Benjamin W. Tatler. *The prominence of behavioural biases in eye guidance. Visual Cognition*, volume 17(6-7):pp. 1029–1054, **2009**.

[ViolaJones2002] P Viola and M Jones. *Robust real-time object detection. J. of Computer Vision*, **2002**.

[WaltherKoch2006] Dirk Walther and Christof Koch. *Modeling attention to salient proto-objects. Neural networks : the official journal of the International Neural Network Society*, volume 19(9):pp. 1395–1407, **2006**.

[Wolfe1989] J M Wolfe, K R Cave, and S L Franzel. *Guided search: an alternative to the feature integration model for visual search. Journal of experimental psychology. Human perception and performance*, volume 15(3):pp. 419–433, **1989**.

[Wu1997] Victor Wu, R Manmatha, and Edward M Riseman. *Automatic text detection and recognition.* In *Proceedings of Image Understanding Workshop*, pp. 707–712. Citeseer, **1997**.

[Yao2012] Cong Yao, Xiang Bai, Wenyu Liu, Yi Ma, and Zhuowen Tu. *Detecting texts of arbitrary orientations in natural images.* In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE*

Conference on, pp. 1083–1090. **2012**. ISSN 1063-6919.

[Yarbus1967] Alfred L Yarbus. *Eye Movements During Fixation on Stationary Objects*, pp. 103–127. Plenum Press, **1967**.

[Ye2005] Qixiang Ye, Qingming Huang, Wen Gao, and Debin Zhao. *Fast and robust text detection in images and video frames. Image and Vision Computing*, volume 23(6):pp. 565–576, **2005**.

[YiTian2011] Chucai Yi and YingLi Tian. *Text string detection from natural scenes by structure-based partition and grouping. Image Processing, IEEE Transactions on*, volume 20(9):pp. 2594–2605, **2011**. ISSN 1057-7149.

[Yin2013] Xu-Cheng Yin, Xuwang Yin, and Kaizhu Huang. *Robust text detection in natural scene images*. **2013**.

[Zhang2008] Lingyun Zhang, Matthew H Tong, Tim K Marks, Honghao Shan, and Garrison W Cottrell. *Sun: A bayesian framework for saliency using natural statistics. Journal of vision*, volume 8(7):pp. 32.1–20, **2008**.

[ZhaoKoch2011] Qi Zhao and Christof Koch. *Learning a saliency map using fixated locations in natural scenes. Journal of vision*, volume 11(3), **2011**.