UNIVERSITÀ
DI TRENTO

Department of Computer Science and Information Engineering

Bachelor's Degree in
Computer Science

Final Dissertation

On Saccade and Human Attention

A model of bottom-up saliency for human saccade simulation

Supervisor
Fausto Giunchiglia

Student
Samuele Conti

Academic Year 2019/2020

Aknowledgements

Thanks to Raffaella, Giancarlo, Giorgia and Benedetta, who supported me in all aspects of life.

Thanks to all the ones that taught me something.

Contents

Abstract

Object recognition is for sure one of the tricky problems that artificial intelligence aims to solve, but, as of today, the task of continuous lifetime learning for object recognition is still far from being resolved. In recent literature (Erculiani, Giunchiglia, and Passerini 2020) a new approach has been proposed to address this problem: it exploits a new concept that uses short videos, *visual objects*, instead of still images, to store the information necessary for the task mentioned.

Along with many advantages, this method introduces also some new challenges, in particular, the challenge of distinguishing when a visual object ends: the moment in which the previous frames are no more *similar* to the current frame; that moment is the one in which we may start seeing a new object, and this is the setting in which we can appreciate the introduction of visual objects into the object recognition field.

The work presented here aims to discuss and build a solid strategy to decide when a real-time video stream has to be split because of the appearance of a new visual object. The main idea is that the saccadic movement of the human eye could be used as a proxy for a shift of attention from one visual object to another. This characteristic movement of the eyes has proved to be deeply related to the deployment of attention, in fact its purpose is to shift the central part of our eye, the fovea, to the most relevant region of the visual stimuli.

To simulate this peculiar shift of attention it's necessary to account for two types of influence: bottom-up and top-down. Top-down influences derive from the internal state of the observer and can be determined by the goals and objectives of the observer himself. Bottom-up influences are independent of those factors, they derive instead from characteristics of the various areas of the visual input that is perceived by the observer.

In this document, I present a model for the estimation of the bottom-up factors that influence saccadic movements. The model is developed to integrate it with a component for top-down influences to obtain finally a complete model of the saccade. Along with the explanation and the implementation of the model I produced an evaluation in comparison with other models for the estimation of bottom-up influences on two different datasets: one for static images and one for dynamic video streams.

1 Introduction

1.1 Is that still my mug?

In our daily life, we encounter a huge quantity of objects and we are able, to some extent, to distinguish them one from the other. Despite being such a fundamental ability for humans, this skill is still far from being implemented via software.

In this work, I will try to approach this problem by adopting the theoretical assumption, derived from Millikan 2000, that there is a distinction between what we perceive and its mental representation. The mental representation is what we call usually *object*, while what we perceive is called *substance*. By definition substances are things '... about which you can learn from one encounter something of what to expect on other encounters, where this is no accident but the result of a real connection . . . '.

In Erculiani, Giunchiglia, and Passerini 2020, it is proposed the hypothesis that all substances that we encounter in our lives are represented in our brain as a series of *visual objects*. Visual objects are short sequences of frames that are *similar enough* one to the other and, given their similarity, we can assume that each frame in the visual object contains the same objects.

In this theoretical framework, the mug from which I drink coffee every morning is stored in my brain as a group of short videos of the mug itself, and in each of these short videos (visual objects) the first frame is similar enough to the last frame, therefore I can be sure that what every frame of the visual object contains my mug.

If we accept this hypothesis, then we must answer a tricky question: how do we define when a frame is sufficiently different from a second one so that it is possible to split the video stream into two separate visual objects?

In Erculiani, Giunchiglia, and Passerini 2020, the solution adopted is to use a similarity measure on the frames of the input video stream: when the similarity is under a certain trained threshold then the algorithm splits the video stream differentiating between visual objects.

Another possible solution would be to compute the regions of interest in an image and assume that each one of them is a different object: when in a video stream the regions of interest change then it means that we are seeing a new object. Instead of the region of interest, we could use some algorithm for background subtraction and apply a similar strategy: when the output of the background subtraction changes we assume to be seeing a new object.

In this work, I am proposing an alternative, biologically inspired, strategy based on the simulation of the saccadic movement of the human eye. The fundamental assumption here is that this particular rapid movement of the eye can be used as an indicator of the shift of attention from an object to another one, and therefore, by simulating the saccadic movement, we can build an algorithm that performs distinction between visual objects in a human-like fashion, exploiting attention shifts as signals to separate different visual objects.

1.2 Structure of the thesis

In this document, I approach the challenge of simulating the human eye's saccadic movement by implementing an algorithm for the computation of visual saliency from bottom-up signals, which is a fundamental component for saccade simulation.

In Chapter 2, I present the topics of saccade and saliency (Section 2.1), starting with a discussion of some historical background (Section 2.2) and then analysing the various aspects that influence saccadic movements (Section 2.3).

Then Chapter 3 presents the model developed, giving both a theoretical explanation (Section 3.1) and a practical step-by-step analysis (Section 3.2).

After that, Chapter 4 evaluates the model proposed in comparison other state-of-the-art models.

Finally, in Chapter 5, after having discussed the results obtained by the model proposed, I underline strengths and weaknesses of the model, and others noticeable observations, while describing the future work.

2 Problem Statement

Eyes don't lie. That may be a saying, but it bears some truth: we can safely state that eye movements carry relevant information regarding the thoughts and goals of the observer: they are considered often as a proxy for shifts of attention. That is the reason why understanding how humans move their gaze is so critical. Over the years, this topic has been studied for various reasons, from marketing (for product placement) to neuroscience (to understand mechanisms that drive attention).

Here I assume that a shift of attention, defined by a saccadic movement of the eye, can be used as a signal to separate one visual object from another. That is why I am proposing an algorithm to estimate visual saliency, which is a fundamental piece for simulating saccadic movements. The following section aims to present the reader with a description of the particular human eye movements that I try to mimic in this work.

2.1 Saccade, scanpaths and fixations

A fundamental fact about human vision is that when we look at a scene, we do not perceive all the visual field[1] in the same way: only a limited area of the eye can capture the image at the maximum resolution. This area in the centre of the eye is called the *fovea.* Outside this small area, which covers around 5° of the whole visual field (approximately 200° to 220°), the quality of vision drops drastically (see Rosenholtz, J. Huang, and Ehinger 2012).

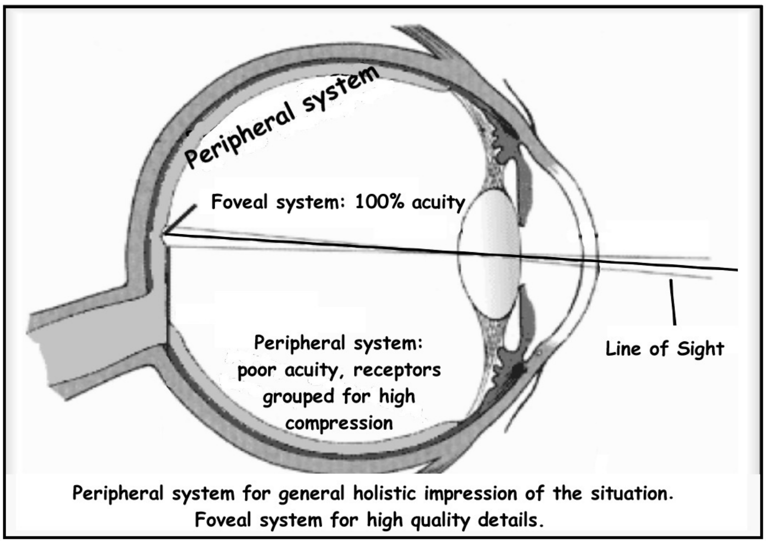
Peripheral system
Foveal system: 100% acuity
Line of Sight
Peripheral system:
poor acuity, receptors
grouped for high
compression
Peripheral system for general holistic impression of the situation.
Foveal system for high quality details.

Figure 2.1: This figure illustrates fovea in the human eye. Source: Hans-Werner34 at English Wikipedia, CC BY-SA 2.0 DE, via Wikimedia Commons.

The reason behind this particular architecture is that an enormous amount of information is captured by our eyes every second, estimated to be in the order of 10^8 or 10^9 bits per second (K. Koch et al. 2006). It would be overwhelming to analyse all this data without any mechanism for data filtering or compression, like the presence of the fovea, helping us process all the incoming information.

One of the consequences of the foveation of human vision is that, when observing a scene, we continually shift our gaze with fast movements (3 to 4 times per second), called *saccades*, that explore

[1] The visual field is the *'spatial array of visual sensations available to observation in introspectionist psychological experiments'* (Wikipedia).

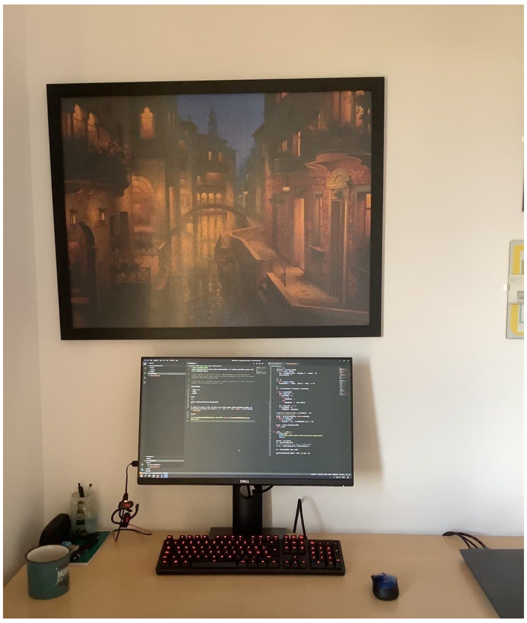

Figure 2.2: Example of a *normal* image, this is how a camera captures the visual field: all the areas of the scene have the same resolution; what the human eye perceives is very different from this.

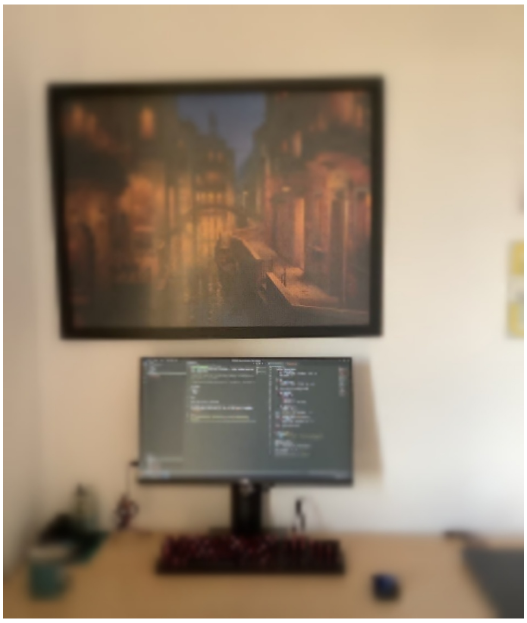

Figure 2.3: Example of a foveated image, this is what the eye actually perceives. The image is generated using code from Zhibo Yang github repository for simulation of foveated vision.

the scene in front of us to move the fovea, the focus of visual attention, to various positions. Because of the foveation of the eyes, we don't have a wholly clear image of the world around us, but, thanks to saccadic movements, we can retrieve multiple high-resolution patches of the whole visual scene and build a consistent internal representation. If we didn't use saccade movement to shift the fovea in this fast and continuous way we would have a clear representation only of the objects that are present in the central part of our visual field, thus greatly limiting our perception of the world around us. Normal vision consists of a continuous rapid sequence of two phases: *saccade* and *fixation*. To sum up, fixation is the name of the period in which the eye remains still on a certain position of the visual field, while saccades are sharp and fast movements that move the fixation point to a new target. The sequence of those movements (fixation-saccade-fixation) over a visual scene is called visual scanpath, or simply scanpath.

Over the last years in neuroscience and neuropsychology, the research around attention mechanisms has shifted from inquiring mental only processes like the model of the mental spotlight, a metaphor of attention on a fixed retinal image from Posner 1980. Lately, more researchers are studying the movements of the eyes as a fundamental piece for the understanding of human attention. For instance, in J. Findlay and Gilchrist 2012, the retinal image is not presented as static photography to the brain of the observer, but it is composed instead of the various scattered pieces of information collected by the fixations directed by the saccadic movements.

The vast majority of the models that aim to simulate the human eye scanpath are based on the concept of *visual saliency* (or simply saliency), namely the property of some regions of the image to attract visual attention. The regions in some image that present the higher saliency are the ones that attract the human gaze with higher probability and therefore there we can find the points in which fixations of a scanpath are usually found.

Since we are trying to exploit shifts of visual attention, to decide where to split visual objects, I am going to discuss here a model for the estimation of bottom-up visual saliency created with the aim to be the first component of a project for the simulation of saccadic movements.

2.2 Historical background

Studies on human gaze behaviour have a long history, dating back to Yarbus 1967 where we find the first approaches used to describe how humans decide which areas of the visual field to direct their attention to. The author studied visual attention in this experimental setting: he showed a painting to different observers while giving them some tasks, for example guessing the age and wealth of the subjects or just freely observing the piece of art. While the observers fulfilled the task, the author kept track of the movements of their eyes and then studied the positions in which observers fixated their gaze. Eventually, he found out that eye movements diverged considerably depending on the given task.

Since then, many other researchers have tried to explain, predict and simulate the positions in which an observer will look in a particular visual scene. This being also strongly connected to where the human observers tend to direct their attention. But defining the concept of attention is not an easy, nor even a solved problem: during this discussion, we'll use as a proxy for visual attention the concept of saliency from the definition in Borji and Itti 2013 reported below.

Definition (Modeling attention)**.** Assume that a set of N images $I = \{I_i\}_{i=1}^{N}$ has been viewed by K human observers.
Let $L_i^k = \{p_{ij}^k, t_{ij}^k\}_{j=1}^{n_i^k}$ be the vector of eye fixations $p_{ij}^k = (x_{ij}^k, y_{ij}^k)$ and their corresponding occurrence time t_{ij}^k for the observer k on the image I_i.
Let n_i^k be the corresponding number of fixations of subject k over i^{th} image.
The goal of attention modelling is to find a function (stimuli-saliency mapping) $f \in \mathcal{F}$ which minimizes the error on eye fixation prediction, that can be expressed as $\sum_{k=1}^{K} \sum_{i=1}^{N} m(f(I_i^k), L_i^k)$, where $m \in \mathcal{M}$ is some distance measure.

2.3 Analysing visual saliency

As of today, it is clear that we can identify mainly two different types of contribution to the mechanisms that move our gaze, namely bottom-up and top-down. I want to underline that this subdivision is quite rough, but a more precise analysis would be tedious and is out of scope for this disquisition. As a reference for a deeper insight on the discrimination of the various factors that concur to determine saliency, I point out Wolfe and Horowitz 2017.

2.3.1 Bottom-up factors

Bottom-up influences are produced by the presence of some feature in the image that attracts the attention of the viewer, for instance, luminance contrast, edge density and colour to cite some possible candidates. This theory was introduced by Treisman back in the 80s (see Treisman and Gelade 1980), with the development of the first model based on the combination of feature maps extracted from the image to create a general saliency map of the whole visual field.

The intuition behind Treisman's theory is that the eyes perceive the image through different channels, dedicated to distinct features. In that case, we can simulate the eyes' behaviour by:

1. taking the input image and applying some decomposition in different feature maps;
2. computing the local importance of each area in every single feature map;
3. recombining all the different feature maps to obtain a summarizing map.

The resulting map stores the influences from the various feature values: we call that a saliency map.

To have an idea of how this mechanism works, take a look at Figure 2.4, where you will notice that there are three bars that *attract your gaze* more than the others: the T-shaped one, the red one and the thicker one. That illustrates in a practical way how bottom-up saliency works: the parts of the image that are different from their surrounding attract more the observer attention. But how can we measure when two areas of the visual input are different? It can be measured using the feature values at the various locations of the scene.

In fact, in years of research, some features have been accurately identified as good predictors of bottom-up saliency; the most influential are colour, motion, orientation and size (see Wolfe and Horowitz 2017).

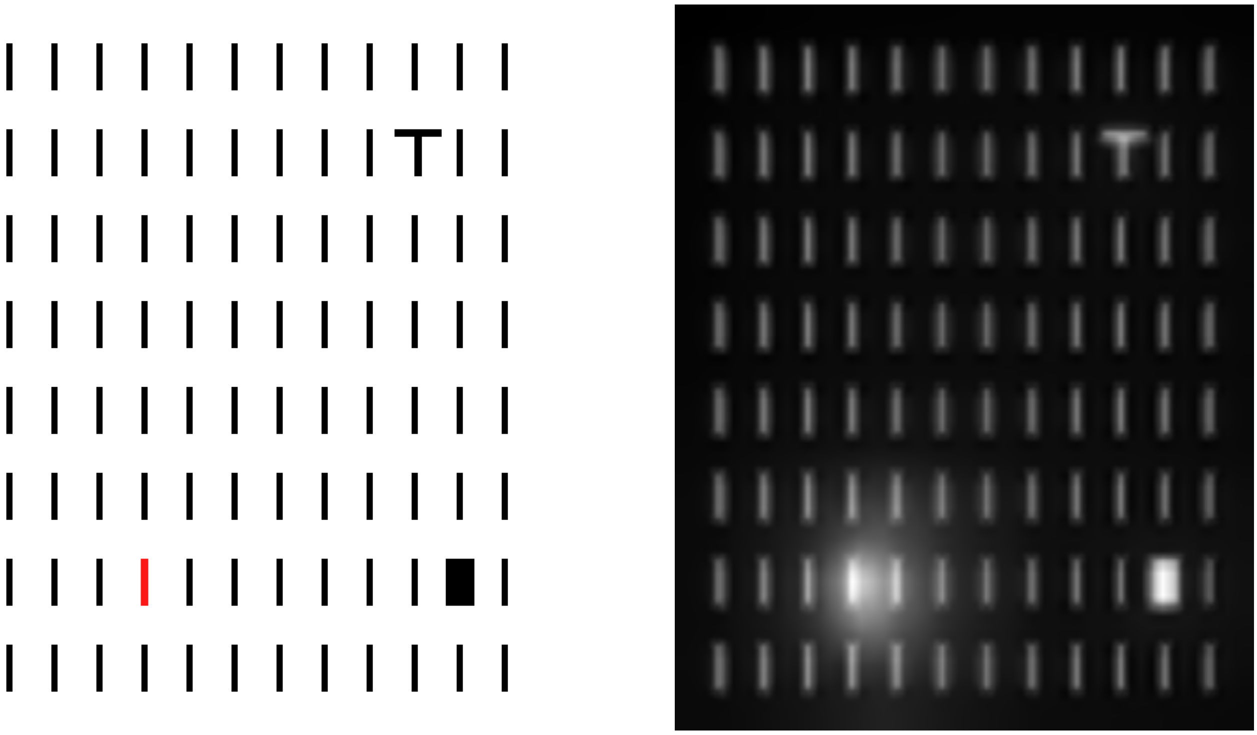

Figure 2.4: The image is an example of how low-level features such as colour, size or orientation can influence the way in which we observe an image by attracting our attention to the locations in which these features present local contrast. On the left you can see the original image, while the one on the right is the saliency map computed by ICOM model.

By looking at the saliency map obtained from the fusion of the various feature maps, you can tell which are the regions of the visual field that *pop out*, which means that capture the attention with more strength, based on their being salient in one or more of the feature maps computed before.

Bottom-up saliency is not only influenced by features and their local differences; a different aspect to take into account is complex spatial statistics that characterize the movement of the eye during free viewing of a scene. Some examples are the tendency of the fixations to cluster at short distances (Engbert et al. 2015; Trukenbrod et al. 2019; Schwetlick et al. 2020) or the mechanism of *inhibition of return* (which is believed to facilitate exploration during visual search, see MacInnes and Klein 2003).

Many models that simulate gaze movements during scene observation have been proposed over the years, both for static images and videos.

Most of these models aim to simulate free viewing, which is often associated with the development of purely bottom-up models because free viewing is considered the task that reduces at minimum top-down influences on the deployment of visual attention.

Some models for saliency estimation, particularly the ones based on Machine Learning techniques, manage to simulate closely human behaviour. But in the case of ML it is often tricky to assume that the model uses only bottom-up features because training a model on a dataset of human fixations introduces the risk to learn also top-down influences that the observer may have experienced while the data was being collected.

2.3.2 Top-down factors

Bottom-up influences are only half of the story. The other source of influences that direct the human gaze derives from top-down cues; these depend on the internal state of the observer, such as tasks given to the observer while viewing the image, like visual search or face recognition.

These influences have been long studied in a mixture of contexts, starting from the execution of daily routine tasks, as in Hayhoe and Ballard 2005, and arriving at artificially built search tasks, as in J. M. Findlay 1997.

It turns out that performing a visual search task is more efficient in terms of search time and fixations count when some properties of the target are known in advance. From J. M. Findlay 1997, we learn that if the observer knows the colour (or shape) of the target of a visual search task, there's a high probability that he will find it within the first few saccades, even with numerous distractors.

In Hayhoe and Ballard 2005 the authors studied the behaviour of the human gaze during the performance of some daily actions, as making a sandwich, and pointed out that the position of fixations collected while carrying out a natural daily task often disagrees with the purely bottom-up saliency of the scene.

It turns out that the majority of saccades while performing one of such tasks are often directed towards a location in a scene *in advance* of some expected event. This behaviour is defined by Ballard just-in-time strategy, which means that observers point their attention to some spots in the visual field exactly before needing the information stored in those locations. Similar results are reported in Land and McLeod 2001, where cricket players look at the bounce point of the ball just before its impact, suggesting that in this way they retrieve important information about the speed and trajectory of the ball to estimate the point of contact with the bat. The last two examples demonstrate how the presence of a task in the observer's mind influences their *gazing behaviour*.

Some interesting observations on top-down influences come from the evaluation of the effects of visual memory in the repeated presentation of the same image. For example, in Kaspar and König 2011, it has been observed that while repeating different presentations of an image not only the observer tends to stare always at the same locations, but also the number of fixated locations decreases. This fact represents a loss of interest in the areas that do not contain interesting information (the ones already explored and considered not interesting).

In the same paper, we can also read that by showing repeatedly the same image to the observers, we can measure an increase in the inter-subject variance of fixation distributions. We deduce that each observer has their interests, resulting in different ways to fixate the scene.

Between the other top-down influences, we find the use of scene structure and meaning, which can give both semantic and syntactic guidance to the observer that has experience of the scene. Previous experience means that the observer has already an internal model or representation, which is exploited to facilitate tasks as visual search (see Wolfe and Horowitz 2017).

Similarly to scene structure and meaning, the two factors of prior history and value of the items have been studied.

The first factor, prior history, influences how the behaviour of the human gaze changes when a scene is (partially or completely) shown before the search experiment: the preview gives the observer some information about the visual scene and this helps him/her obtain a better performance during the search task (the observer finds the searched object faster and with a smaller number of saccades).

The second factor, the value of the items, influences the gazing behaviour depending on the value of the reward that the users gets when they complete the search task: if some feature characterizes objects that give a better reward in a search task, the observer demonstrates higher sensibility to that feature, even after some months.

2.4 Spatial and spatiotemporal saliency

The vast majority of models developed to simulate human gaze behaviour are based on the concept of *saliency map*, namely a map of an image where each pixel has a value of saliency: the more an area is salient, the more it attracts the human gaze.

The simplest way to compute a saliency map is to pick an image, record some people looking at it for a certain amount of time and track the points where the observers' gaze fixates: the coordinates of those points constitute the experimental data. The *spatial saliency* is obtained by computing the 2D density of the gaze points over the space of the image.

Basically, to get a spatial saliency map you have to record some observers while looking at an image and keep track of the image areas that attract their gaze. A detailed example of this procedure is presented in Le Meur et al. 2006.

What I wrote in the previous lines is the standard way to compute saliency for images; by extending that procedure to videos we obtain what is known as *spatiotemporal saliency map*. The difference between a spatial and a spatiotemporal saliency map is that the first is computed from the observation of an image, a static visual input, where the second is derives from dynamical stimuli, namely a video.

Given those two definitions of saliency map, in the literature exist examples of both studies of spatial and spatiotemporal saliency; while the last one, spatiotemporal saliency, has the characteristic of being closer to our daily experience in a dynamic world, the first, spatial saliency, has been studied for a long time since it guarantees a simpler environment to analyse the effect of the various features and top-down influences. Interestingly, some of the most successful models created to get spatial saliency out of images have been adapted and redesigned to perform the computation of spatiotemporal saliency, for example Itti, C. Koch, and Ernst 1998 and Itti, Dhavale, and Pighin 2004. That is the case also for the model I am going to present.

3 ICOM model

The model I decided to implement is a model that accounts for **bottom-up influences** in a dynamic setting to compute the **spatiotemporal saliency** of each frame of a given input video stream. I decided to use as starting point a simple and highly parallelizable model presented in Itti, C. Koch, and Ernst 1998, which has had a long-lasting impact in the literature of bottom-up models for visual saliency.

Part of my contribution consists in the development of a highly parallelized Python implementation of this starting model, with the major improvement of expanding it for the computation not only of spatial but also of spatiotemporal saliency. I obtained this result by creating a motion detection filter which adds a dynamic component to the spatial saliency estimation provided by the reference model.

The model has been developed to create a simple and fast implementation of a bottom-up visual saliency estimator. During the development of the model, particular care was used to produce an algorithm that facilitates a future integration with a top-down saliency estimator to obtain, in the end, an algorithm for the simulation of saccadic movements.

The model is called ICOM from Intensity, Colour, Orientation and Motion: it will soon be clear why.

3.1 Theoretical basis

The model is based on biologically plausible observations, which belong to the paradigm of feature integration theory (FIT), presented in Treisman and Gelade 1980. The strategy is to take visual input and extract different feature maps for a fixed set of features; these are colour, orientation and intensity for the original model, to which I added motion.

The fundamental assumption behind FIT is that each location in the image competes with all other locations to attract the human gaze; the conspicuity, or saliency, of a location, determines its importance and is defined by the local values of all the features: the locations in which some feature value *pops out* with respect to the neighbouring area are the most salient.

From this assumption derives that between neighbouring locations there is a competition for saliency: neighbouring locations compete for saliency in all the different feature spaces.

It can be imagined as if there is only a certain amount of saliency to be allocated for the motion feature and all the regions of the image are competing to get the maximum possible saliency in the motion feature space. The same is true for all the other features. That is called *intra-feature* competition.

Once we have a saliency map for each feature, we need to combine them to get the global saliency map that accounts for all bottom-up influences. In this step, the locations compete across different feature spaces, generating the phenomena of inter-feature competition.

As said before, the reference model was built to account only for spatial saliency; here it has been expanded by inserting an algorithm for motion detection and, therefore, a feature map for motion, which will be discussed later.

3.2 ICOM algorithm

To get the gist of the working of the algorithm, in Algorithm 1 the pseudocode for the ICOM model is presented, it will be used as a basis to guide the in-depth analysis presented in the following sections.

Algorithm 1: getSpatiotemporalSaliency(Frame *previous*, Frame *current*)

```
   Result: Spatiotemporal saliency map derived form the two subsequent frames
 1 /* extracting feature signal */
 2 intensityFeatureMap = extractIntensityMap(current)
 3 colorSpaceRG, colorSpaceBY = extractColorMaps(current)
 4 orientationFeatureMap = extractOrientationMap(current)
 5 motionFeatureMap = extractMotionMap(previous, current)
 6
 7 /* computing conspicuity, each feature in its own way */
 8 intensityConspicuityMap = intensityConspicuity(intensityFeatureMap)
 9 colorConspicuityMap = colorConspicuity(colorSpaceRG, colorSpaceBY)
10 orientationConspicuityMap = orientationConspicuity(orientationFeatureMap)
11 motionConspicuityMap = motionConspicuity(motionFeatureMap)
12 conspicuityMaps = [intensityConspicuityMap, colorConspicuityMap,
     orientationConspicuityMap, motionConspicuityMap]
13
14 /* $\mathcal{N}$-normalization of conspicuity maps */
15 foreach map $\in$ conspicuityMaps do
16     $\mathcal{N}$-normalize(map)
17 end
18
19 /* sum of all conspicuities and $[0;1]$ normalization */
20 saliencyMap = sumAll(conspicuityMaps)
21 /* saliency map may or may not be normalized in range $[0;1]$ before returning */
22 saliencyMap = simpleNormalization(saliencyMap)
23 return saliencyMap
```

3.2.1 Feature maps

First, the algorithm takes the visual input and extracts the following feature maps:

- intensity;
- colour;
- orientation;
- motion.

The first signal to be extracted is intensity: it is computed as the average of the three-colour channels for every single pixel as from Equation (3.1).

$$\mathcal{I} = (r + g + b)/3 \tag{3.1}$$

Then, according to Line 3 of the pseudocode, we extract the colour signal. In the visual cortex, colours are represented by a colour *double-opponent system* (see Hurlbert 2003), therefore we model colour feature map as two different maps which account for the contrast between opponent colours. The first map, $\mathcal{RG}$, accounts for the opposition of red and green colour channels as shown in Equation (3.2), where R is the channel for red, obtained from Equation (3.3) and G is the channel for green, from

Equation (3.4).

$$\mathcal{RG} = R - G \tag{3.2}$$
$$R = r - (g + b)/2 \tag{3.3}$$
$$G = g - (r + b)/2 \tag{3.4}$$

In the same way we can define the second colour opponency channel for blue and yellow, $\mathcal{BY}$, as shown in Equation (3.5).

$$\mathcal{BY} = B - Y \tag{3.5}$$
$$B = b - (r + g)/2 \tag{3.6}$$
$$Y = (r + g)/2 - |r - g|/2 - b \tag{3.7}$$

All negative values in the colour maps presented are set to zero.

The next signal to be extracted is the orientation feature map (pseudocode at Line 4) from the input image intensity. For this extraction we use Gabor filters, powerful tools used in computer vision to enhance particular orientations in an image.

In the algorithm, we use four filters to cover the following orientations: 0°, 45°, 90° and 135°. Gabor filters have the advantage of approximating the receptive field sensitivity profile (impulse response) of orientation-selective neurons in the primary visual cortex (Bensmaia et al. 2008).

The last step is the extraction of motion feature: in this case, I decided to use an algorithm from the fast block-matching motion estimation literature, particularly the Diamond Search (DS) motion estimation algorithm presented in Shan and Kai-Kuang 2000. This algorithm is both accurate and highly efficient if compared to other algorithms of the same family.

This approach presented a challenge: since Diamond Search is an algorithm for motion estimation (slightly different from motion detection), I had to implement some modifications to the original code to adapt it to its new use case.

To get a gist of the way this algorithm work, I provide a quick and rough explanation, which won't be enough to fully explain the algorithm and its advantages: for this purpose, I suggest reading DS original paper, Shan and Kai-Kuang 2000, or the subsequent survey on block-matching motion estimation algorithms Y.-W. Huang et al. 2006, that illustrates the advantages that DS provides compared to other algorithms of the same family.

The Diamond Search algorithm is used to estimate the motion between two subsequent frames of a video: it divides the image into blocks of size $N \times N$ and computes the movement (as a rigid translation) of every single block between the two frames.

To get the displacement of a single block the DS algorithm performs a search among neighbouring blocks to find the one that best matches the starting block. When two blocks have been coupled as start and stop of the transition, the vector that links the two is the motion estimation: it is called motion vector and represents the estimated shift of the starting block.

The difference between any two blocks corresponds to the sum of absolute differences between the corresponding pixels of the two blocks.

3.2.2 Multiscale centre-surround differences

The feature maps described above are the starting point, the next step of the algorithm is performed for the sake of compliance with what we know about the human receptive system. For each one of the feature maps, it creates a Gaussian pyramid that consists of the map at nine different resolutions, from scale 0 to scale 8, meaning that the first map of the pyramid will be in a proportion of 1 : 1 with the original image, while the last will be in proportion 1 : 256.

The purpose of this is to compute the feature maps, not as artefacts obtained by simply filtering the original image, but, instead, as centre-surround differences applied to those filtered artefacts. Since typical visual neurons are most sensitive in the centre, while stimuli from the surrounding concentric region inhibit the neuronal response, an architecture that exploits centre-surround differences is a good approximation of visual perception. The resulting mechanism is good for detecting, with a biologically plausible system, local spatial discontinuities, and has been demonstrated to share similarities with the information processing of the lateral geniculate nucleus (LGN) and primary visual cortex of the human brain, see Itti, C. Koch, and Ernst 1998.

To implement such a system, each feature map is duplicated at different scales (the nine scales of the Gaussian pyramid), obtaining a list of nine partial feature maps, each one at a different resolution. To get the final complete feature map from each Gaussian pyramid we perform several centre-surround differences, implemented as concrete differences between different layers (fine and coarse scales) of the Gaussian pyramid for every single feature.

The finer scale represents the centre, while the coarser scale represents the surround; for each pixel of the centre, at a scale $c \in \{2, 3, 4\}$ we can pick a pixel from a coarser scale $s = c + \delta$ that corresponds to the surround, where $\delta \in \{3, 4\}$.

Algorithm 2: multiscaleDifferece(Map $featureMap$)

```
   Output: Spatiotemporal saliency map derived form the two subsequent frames
1  /* creating the Gaussian pyramid */
2  gaussianPyramid = getGaussianPyr($featureMap$)
3  /* the pyramid is a list of 9 maps */
4  /* from scale 1:1 (original scale) to 1:256 (coarser scale) */
5
6  /* computing multiscale differences */
7  multiscaleDifferences = [ ]
8  foreach $c \in \{2, 3, 4\}$ do
9      foreach $\delta$ in $\{2, 3\}$ do
10         s = c + $\delta$
11         center = gaussianPyramid[c]
12         surround = interpolate(gaussianPyramid[s], center.shape)
13         multiscaleDifferences.append(center-surround)
14     end
15 end
16 return multiscaleDifferences
```

By varying both c and δ we can compute multiscale differences which include different size ratios between the centre and surround regions of the image. Naturally, to make a point-by-point subtraction between the two scales, we need to use some sort of interpolation: in this case, linear interpolation to the finer scale, namely c, is performed. The pseudocode for this algorithm is presented in Line 2.

Finally, to get the complete feature map, we resize each set of centre-surround differences to scale 4 and sum them with point-by-point addition. This is the penultimate step to obtain the saliency map: the algorithm needs to perform also the addition of conspicuity between different features, which in turn needs a mechanism for normalization.

To be clear, I report here an example. Let's pick the intensity feature map we described before in Section 3.2.1: we compute an intensity Gaussian pyramid of nine levels where each level corresponds to a different resolution of the same intensity map obtained from Equation (3.1). To get the final feature map for intensity, we compute

$$\mathcal{I}(c, s) = |I(c) \ominus I(s)| \tag{3.8}$$

where $\ominus$ stands for the multiscale point-by-point difference with interpolation to the lower scale and where, as written previously, $c \in \{2, 3, 4\}$, $s = c + \delta$ and $\delta \in \{3, 4\}$.

$\mathcal{I}(c, s)$ is a list of partial maps containing centre-surround differences between finer and coarser scales of the intensity feature map. To get the conspicuity map for intensity (pseudocode Line 8), we need to sum all those scaled maps and get the general multiscale difference, and then normalize the resulting map to apply intra-feature competition. Only after the normalization step, we can join the different conspicuity maps.

3.2.3 Joining different features

Once the conspicuity map for each feature is computed we need to join all of them and obtain the saliency that summarizes all the influences from the different features in one single map.

As said before, addition between the partial conspicuity maps is performed with interpolation to scale 4 and point-by-point addition, but there's still the need to normalize each feature conspicuity to enable a comparison between them and to implement intra-feature competition.

The normalization operation (Algorithm 1, Line 16) used is created with the purpose to promote the feature maps that contain a few strong peaks of activity (salient areas) and suppress those characterized by a lot of comparable activity responses.

In order to obtain this result Itti, C. Koch, and Ernst 1998 proposes a normalization operator $\mathcal{N}$ which works as follows:

1. the feature map is normalized in a range $[1...M]$ to eliminate modality-dependent amplitude differences;

2. given the global maximum M we compute the average of all the local maxima $\overline{m}$ computed over blocks of a fixed size;

3. the map is globally multiplied by $(M - \overline{m})^2$.

This type of normalization is grounded in the finding that, in the human brain, we find some anatomically defined connections that inhibit neighbouring activation signals enabled by the same visual feature (see Cannon and Fullenkamp 1996). That is precisely what in Section 3.1 is called intra-feature competition.

From the normalization operator $\mathcal{N}$ we obtain that in maps in which the local maxima are closer to the global maximum we tend to decrease the importance of the whole feature map, and vice versa, in maps in which the global maximum of conspicuity is much stronger than the average local maxima we promote the map value since it contains a location which strongly *pops out* from the rest.

To explain it more clearly, if an image has bright colours the intensity map will have high values in pretty much all its regions, so we don't want intensity to be a strong factor for saliency detection: if all regions are bright then none of them will *pop out* for its brightness. If in the same image we find everywhere the same colour and intensity but a small region in which the colour while remaining bright, is different, then the colour feature (actually colour contrast) will yield more information than intensity, therefore the algorithm will promote conspicuity given by the colour feature maps.

This process is explained in 3.1, where the original image presents bright colours everywhere. Since the signals for the intensity feature map are strong at every location, none of the regions locally stands out, hence conspicuity from the intensity feature map is suppressed by the $\mathcal{N}$ normalization operator.

At the same time the area where the colour change is very salient in the colour-contrast feature map, which has only one big absolute maximum activation signal in that area, therefore its conspicuity response is reinforced by $\mathcal{N}$. The result is that saliency is all concentrated in the region salient in colour-contrast feature space.

The normalization operator, regularizing the feature maps depending on the signal values, implements also the inter-feature competition that strengthens the responses for the feature channels that present high local contrast at the expense of weakening the response of the feature maps that do not present this property.

Now that the normalization operator is explained, we can proceed and see how the conspicuity maps for the various features are obtained and then combined.

For the conspicuity map of intensity, which has been presented before, refer to Equation (3.8); the normalized conspicuity map for intensity $\overline{\mathcal{I}}$ is computed as:

$$\overline{\mathcal{I}} = \bigoplus_{c=2}^{4} \bigoplus_{s=c+3}^{c+4} \mathcal{N}(\mathcal{I}(c,s)) \tag{3.9}$$

Where the symbol $\bigoplus$ represents the across-scale addition. The scales are c for centre and s for surround, obtained as described previously (in Section 3.2.2).

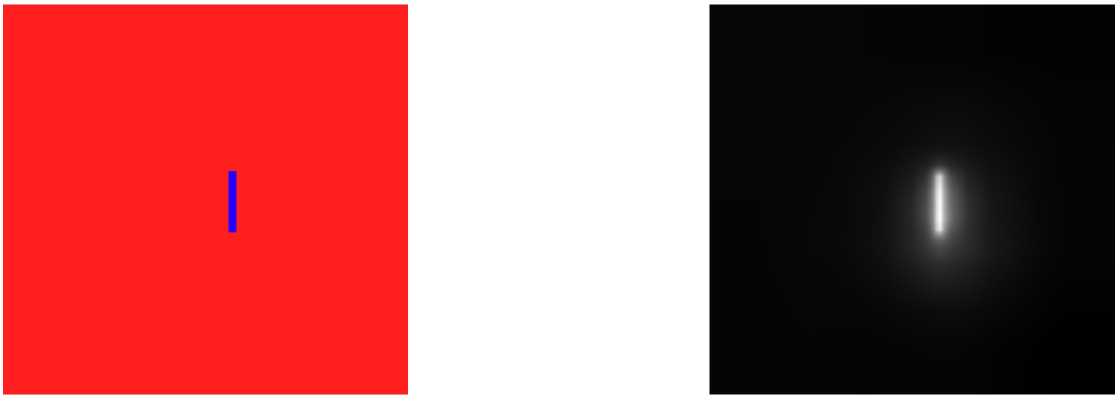

Figure 3.1: On the left the original image, on the right the final saliency map obtained with ICOM.

The colour conspicuity map $\mathcal{C}$ is trickier to compute because the centre-surround difference is computed alongside the opponent colours difference. We define centre-surround differences for the $\mathcal{RG}$ and $\mathcal{BY}$ channels in the following way

$$\mathcal{RG} = |(R(c) - G(c)) \ominus (G(s) - R(s))| \tag{3.10}$$

$$\mathcal{BY} = |(B(c) - Y(c)) \ominus (Y(s) - B(s))| \tag{3.11}$$

Therefore, also the computation of the normalized colour conspicuity map $\overline{\mathcal{C}}$ is a little different, as shown in Equation (3.12).

$$\overline{\mathcal{C}} = \bigoplus_{c=2}^{4} \bigoplus_{s=c+3}^{c+4} [\mathcal{N}(\mathcal{RG}(c,s)) + \mathcal{N}(\mathcal{BY}(c,s))] \tag{3.12}$$

Orientation conspicuity $\mathcal{O}$ is computed for several orientations. Given θ as the variable that represents the orientation, we get the orientation centre-surround maps by computing Equation (3.13) and then, the normalized orientation conspicuity $\overline{\mathcal{O}}$ from Equation (3.14).

$$\mathcal{O}(c,s,\theta) = |O(c,\theta) \ominus O(s,\theta)| \tag{3.13}$$

$$\overline{\mathcal{O}}_{\theta\in\{0,\frac{\pi}{8},\frac{\pi}{4},\frac{3}{8}\pi\}} = \mathcal{N}(\bigoplus_{c=2}^{4} \bigoplus_{s=c+3}^{c+4} \mathcal{N}(\mathcal{O}(c,s,\theta))) \tag{3.14}$$

Finally, for the motion feature the algorithm uses the same mechanism used for intensity, therefore the normalized conspicuity of the motion map $\overline{\mathcal{M}}$ is computed as in Equation (3.15).

$$\overline{\mathcal{M}} = \mathcal{N}(\bigoplus_{c=2}^{4} \bigoplus_{s=c+3}^{c+4} \mathcal{M}(c,s)) \tag{3.15}$$

3.2.4 Saliency map

Once we have normalized all the conspicuity maps from the different feature spaces, weighting each feature based on the local contrast of its responses, we can add them all and obtain the final saliency map. Usually, this saliency map is then normalized again so that each value is in the range $[0,1]$, see Algorithm 1 at Line 20 and Line 22.

A schema with the general summary of the whole process is reported in Figure 3.2.

4 Evaluation

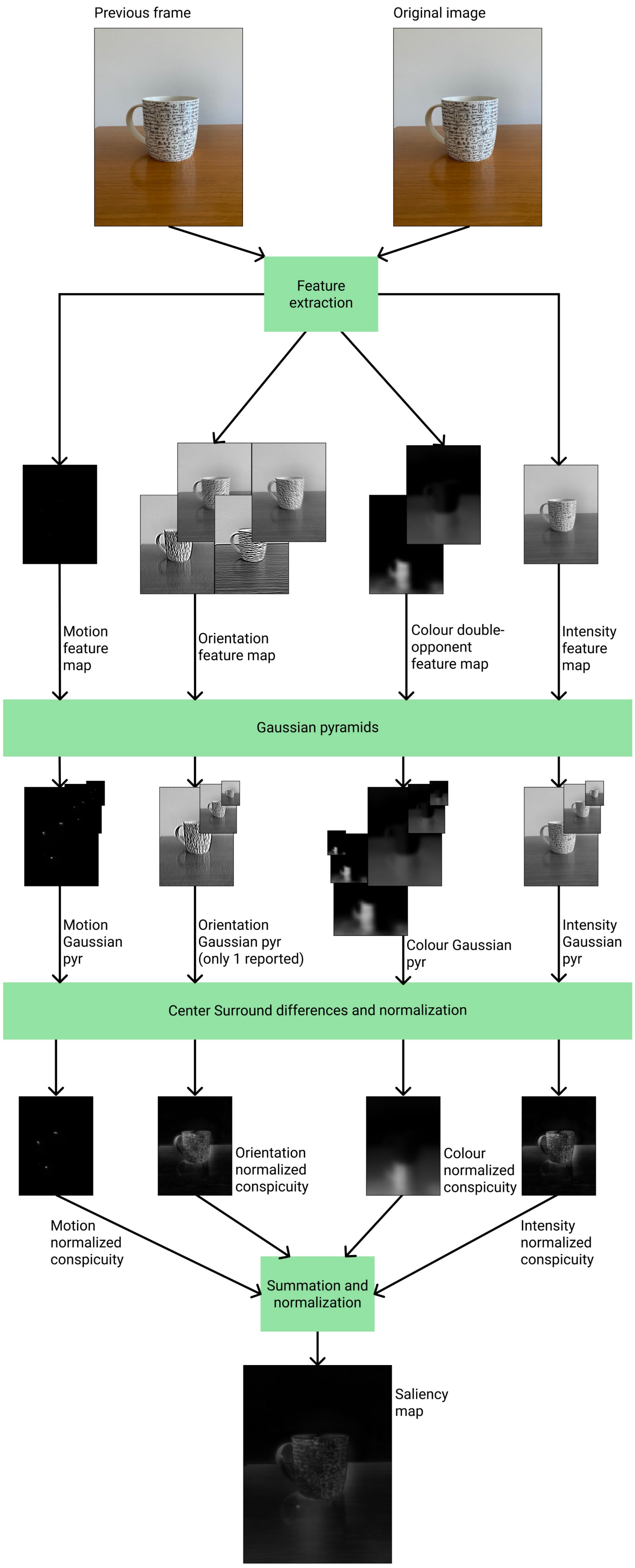
Previous frame
Original image
Feature extraction
Motion feature map
Orientation feature map
Colour double-opponent feature map
Intensity feature map
Gaussian pyramids
Motion Gaussian pyr
Orientation Gaussian pyr (only 1 reported)
Colour Gaussian pyr
Intensity Gaussian pyr
Center Surround differences and normalization
Orientation normalized conspicuity
Colour normalized conspicuity
Motion normalized conspicuity
Intensity normalized conspicuity
Summation and normalization
Saliency map

Figure 3.2: This figure illustrates the complete flow that the ICOM algorithm performs, starting from two frames (current and previous) and arriving at the saliency map.

4.1 Models

The models selected for a comparison with the algorithm proposed here are three:

- RAND, a dummy model that always outputs a random uniform distribution of saliency.

- ITTI, from Itti, C. Koch, and Ernst 1998, represents the baseline, the model on top of which the algorithm in this document builds;
- PQFT, from C. Guo, Ma, and Zhang 2008, a more recent and performing model;

Of all the models used in this work, a Python implementation has been developed and is publicly available on my GitHub page[1].

4.1.1 RAND model

This model outputs only a saliency map in which every pixel has a saliency value defined by a random uniform distribution in the range $[0, 1]$. This model is used as a base reference to measure how much the other models are better than a random prediction for image saliency.

4.1.2 ITTI model

This is the model used as the basis to create the algorithm proposed in this paper. It has been developed to estimate the saliency of static images (spatial saliency) through the combination of feature signals collected through various channels, following the already cited feature integration theory. The model takes into account the following features: colour, intensity and orientation. The different spatial locations compete for saliency in each of the feature spaces, the locations that locally stand out from their surroundings are the ones to get a higher level of activation in the feature map. Then, all the feature maps are combined to form a unique saliency map.

The original implementation of this model does not account for motion, which means that it can account, as already mentioned, only for spatial saliency and not for spatiotemporal saliency; this is the limitation that the algorithm proposed, ICOM, aims to overcome. Therefore, the choice to use ITTI model as a competitor in the evaluation is due to the intention to measure how much saliency estimation can improve by taking into account the motion feature in the domain of spatiotemporal saliency estimation.

Since ITTI model was developed for static images, but we use also a dataset of videos for the evaluation, in the case of spatiotemporal saliency, it has to be run on each video frame to get the estimated saliency comparable to the saliency deriving from human fixations.

4.1.3 PQFT model

This model exploits the phase spectrum of the Fourier transform of the image, which is used to discover the image locations that carry more information in the sense of being different from their local surroundings.

PQFT model uses a quaternion Fourier transform where the value of each pixel is represented as a quaternion composed of intensity, colour (two double-opponent channels) and motion. Using these features, the model is able to estimate spatiotemporal saliency; since the computation of the Fourier transform of the phase spectrum is computationally light, the model is not only one of the most accurate but also one of the fastest in the saliency estimation field.

PQFT model has been chosen for the evaluation precisely because it achieves some of the highest scores in saliency estimation tasks, excluding the ones based on machine learning techniques.

Being a model realized for spatiotemporal saliency estimation, no special care has to be used while comparing it to the ICOM model.

4.1.4 ICOM model

The model proposed here has been widely discussed in the previous pages, so we point out only that this model an extension of ITTI model which adds a component for motion estimation. Therefore, evaluating the algorithm's accuracy on static images with ITTI model and with ICOM model produces the same results. For this reason the results of ICOM are not reported when analysing spatial saliency; only ITTI model result are reported as they represent both the models.

4.2 Datasets

The ICOM model can be used both on static images and on videos, hence I decided to use a dataset of images to evaluate performance in the case of spatial saliency estimation and a dataset of videos

[1] https://github.com/Samaretas?tab=repositories

to evaluate performance in the spatiotemporal saliency domain. In the following sections, we present the datasets and their characteristics, then, in Section 4.3, we discuss the practical procedures used to measure the performance of the multiple algorithms presented before.

4.2.1 Dataset for spatial saliency

The dataset chosen to evaluate spatial saliency is the MIT/Tuebingen Saliency Benchmark (from Judd, Durand, and Torralba 2012; Borji and Itti 2015), which consists of 1003 images with heterogeneous content and context: natural, fractal, social, indoor, outdoor, action, noisy and many other scene categories, along with annotations of fixation positions from more than 20 observers.

In practical terms, the dataset contains a collection of images and the corresponding fixation maps that have been created while the observers were watching the images. Each observer has been shown each image for a span of 3 to 5 seconds, allowing the recording of an amount of 16 fixation points on average for each pair image-observer. For every observation, there is a map where the fixated points are set to 1, while all the other points are set to 0. That is what we call fixation map, or GSM^F. In the evaluation procedure, we need another map different from the fixation map, which is the ground-truth saliency distribution map GSM^D, which is a map that represents density distribution of saliency in the image and is obtained by convolution of GSM^F with a Gaussian filter and then normalization in the range $[0, 1]$.

For the data collection, the observers were instructed to freely view the images, reducing the influence of top-down cues in the selection of fixation positions.

4.2.2 Dataset for spatiotemporal saliency

The dataset used for the evaluation in the spatiotemporal domain is DHF1K (presented in Wang, Shen, F. Guo, et al. 2018; Wang, Shen, Xie, et al. 2019), a dataset consisting of a collection of 1000 videos and annotations of fixation and saccades of 17 different observers. The subjects were instructed to freely observe the videos, meanwhile their eye position has been tracked with an eye-tracking device to measure the position of their gaze. In this way, the authors collected the information necessary to get a fixation map GSM^F and a saliency distribution map GSM^D for each frame of the video stimuli.

Videos in the dataset are taken from a large variety of settings and possible environment, with special care to the presence of relevant features that influence this kind of task, namely camera and objects motion, lighting, presence of humans or animals in the images etc. All the videos have the same resolution (640x360 pixel) and fps (30 fps), but they differ in length.

The dataset contains both fixation annotation for each video frame and the corresponding saliency distribution, obtained via convolution with a Gaussian filter on the fixation maps. Due to issues with the processing time, only 50 out of the 1000 videos of the dataset are used for the evaluation.

4.3 Measures

Now that we have chosen the algorithms and the datasets, we have everything we need to obtain a saliency map for a given image or frame of a video recording. In this section, we present the measures used for the comparison with the aforementioned models.

Since saliency is a way to estimate interesting areas in some visual input, may it be spatial or spatiotemporal, the most straightforward way to evaluate a model is to compare the saliency map created by the algorithm, called also estimated saliency map (ESM), with the saliency map computed using experimental data, namely ground ground-truth saliency map (GSM).

Note that both ESM and GSM are topographical maps of some image where each pixel has a value s (usually normalized in the range $0 < s < 1$) that represents how much that pixel is salient.

In the literature, there is a long list of measures used to test the adherence of ESM to GSM (see Bylinskii et al. 2019 for an exhaustive discussion), and there seems not to be general agreement on one standard measure, so here I decided to use the most important ones:

- **AUC**;
- **Kullback-Leibler divergence**;
- **Line correlation coefficient**;
- **String edit distance (Hamming)**.

4.3.1 AUC

The Area under Receiver Operating Characteristic (ROC) curve is the most common among accuracy measures for saliency estimation algorithms. This measure is frequently used in the field of machine learning because it is useful in evaluating the performance of a classifier; in our case to use this measure we assume ESM to be a classifier for each pixel which discriminates *salient* vs. *non-salient* classes.

The ground truth for the classification is the map of fixations, GSM^F: each fixated pixel is considered salient and vice versa. By varying the threshold value that we use on the ESM to define salient and non-salient points, we can compute the ROC values for the ESM; AUC represents the ratio

$$\frac{TruePositiveRatio}{FalsePositiveRatio}$$

namely the Area under the Receiver Operating Characteristic curve computed at the different threshold values. Here the *True Positives* are the points classified as salient in ESM (above the threshold) and as fixated in GSM^F, while the *False Positives* are the ones above the threshold in ESM classified as salient that are not fixated in GSM^F.

The *True positive Ratio* is the ratio of true positives to the total number of fixations, the *False Positive Ratio* is the ratio of false positives to the total number of saliency map pixels at a given threshold.

The maximum value for this metric is 1: the higher the score, the more accurately ESM predicts GSM.

Spatial Saliency

To get the AUC score for a static image, we need to have the ESM computed by the model and, as GSM, we use the fixation map, namely GSM^F. GSM^F is used as ground truth of the classification where fixated points are positive samples and non-fixated points are negative samples. The ESM is treated as a classifier by varying the threshold saliency value that defines if a point is estimated as salient or not; all the points classified as salient in ESM and fixated in GSM^F are true positives, and so on.

We compute the ROC values at various thresholds and then use this result to draw the Area Under ROC curve. The mean scores for the AUC metric are shown in Figure 4.1.

Spatiotemporal Saliency

To adapt this measure to the spatiotemporal setting we can efficiently compute AUC for each frame of the video stimuli and then average all the AUC scores over the whole video. In Figure 4.2 is shown the result of the AUC measure applied to the four algorithms.

4.3.2 Kullback-Leibler divergence

The KL-divergence, also known as relative entropy, is a measure commonly used to compute the difference between distributions: differently from AUC, here, the ground truth is a distribution of saliency, not a classifier for salient vs non-salient locations. Therefore, to compute KL-divergence, we use GSM^D instead of GSM^F.

With Kullback-Leibler divergence, we measure the distance between the distribution of estimated saliency and the distribution of ground-truth saliency. If i stands for a single pixel in the image, then KL-divergence is computed as

$$KL(ESM, GSM^D) = \sum_i GSM_i^D \log\left(\left(\varepsilon + \frac{GSM_i^D}{\varepsilon + ESM_i}\right)\right) \tag{4.1}$$

where ε is a small regularization constant. Note that this is not the only strategy to evaluate saliency estimation with KL-divergence, but it's the most common (see Bylinskii et al. 2019).

KL-divergence measures the loss of information that derives from using ESM to approximate GSM^D and, being a measure of divergence between the two distributions, the goal of a saliency

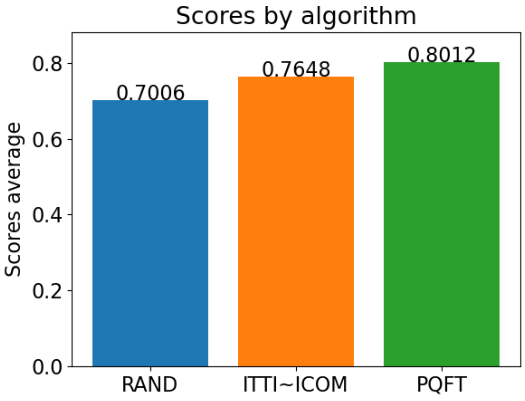

Figure 4.1: The AUC score computed on the image dataset for the three models (ITTI and ICOM are equivalent in the spatial domain). The higher the AUC score, the better the model predicts saliency.

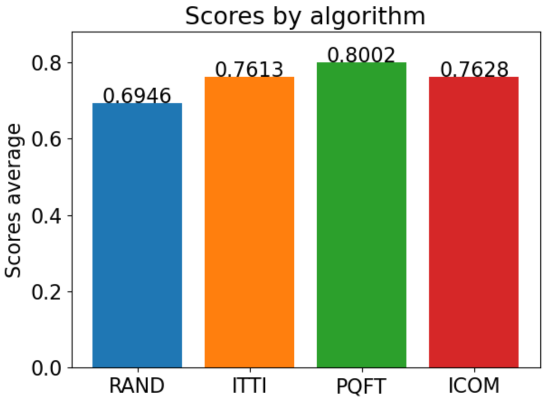

Figure 4.2: The AUC score for the four different models averaged over the first 50 videos in the dataset. A higher score indicates better performance in saliency prediction.

model is to get the *lowest possible KL score*. A lower KL score means that the estimated saliency map has a low divergence from the ground truth.

Spatial Saliency

For spatial saliency, the computation is straightforward: we need only to compute GSM^D for each fixation map and apply Equation (4.1) to compare it with the ESM. In Figure 4.3 are illustrated the scores averaged over the whole image dataset.

Spatiotemporal Saliency

To apply this measure to the spatiotemporal domain, we can, as for the AUC measure, get an ESM for each frame, then compare it with the GSM for the corresponding frame.

In the end we have a KL score for each frame: we just need to average KL scores to get a measure of the distance of spatiotemporal saliency between estimated and ground-truth in a single video; results are shown in Figure 4.4.

4.3.3 Linear correlation coefficient

Also known as *Pearson correlation coefficient*, this is another important measure in the visual saliency field; it computes the linear correlation between two distributions, ESM and GSM^D in this case, and returns a scalar value between -1 and 1: the closer the return value is to 1, the stronger the correlation is between the two maps. If the value is close to 0, then the maps do not present any correlation, while, if the value is negative, they show inverse correlation.

Since this measure uses distributions we need to filter GSM with a Gaussian convolution, as for KL-divergence. Once we have both distributions ESM and GSM^D we compute μ_{GSM^D} and $\sigma^2_{GSM^D}$ as mean and variance of GSM^D map and μ_{ESM} and σ^2_{ESM} as mean and variance of ESM map, then linear correlation coefficient (LCC) is computed as

$$LCC(GSM^D, ESM) = \frac{\sum_{x,y}(GSM^D(x,y) - \mu_{GSM^D}) \cdot (ESM(x,y) - \mu_{ESM})}{\sqrt{\sigma^2_{GSM^D} \cdot \sigma^2_{ESM}}} \tag{4.2}$$

This measure has the property of being confined in the interval $[-1, 1]$, therefore it gives a clear and absolute evaluation on how much the ESM is efficient in predicting real saliency.

As mentioned before, this algorithm aims to get a value that expresses the strength of the correlation between ESM and GSM^D: the closer this value is to $+1$, the better ESM predicts saliency.

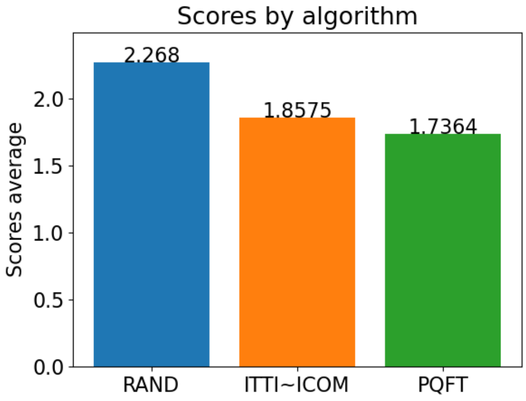

Figure 4.3: The KL-divergence for the three different models (ITTI and ICOM are equivalent in the spatial domain); the score is averaged on all the images of the MIT/Tuebingen Saliency Benchmark dataset. The lower the KL score, the better the model predicts saliency.

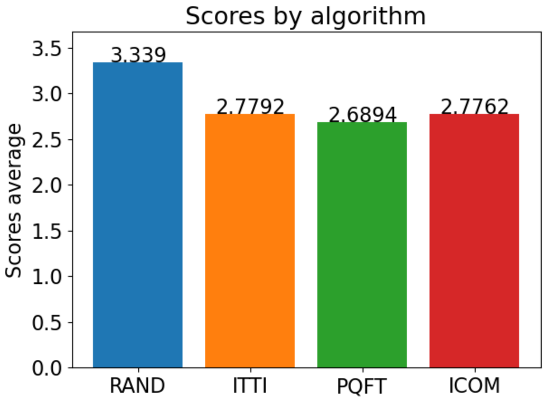

Figure 4.4: The KL-divergence for the four different models; the score is averaged on the first 50 videos of the dataset. The lower the KL score, the better the model predicts saliency.

Static

Each image is processed with the algorithm to obtain ESM then, every fixation map is convolved with a Gaussian filter to retrieve a ground-truth distribution of saliency GSM^D. Results of the evaluation with LCC measure are summed up in Figure 4.5.

Dynamic

Linear correlation coefficient is a measure that finds a natural extension to the spatiotemporal domain by computing the LCC-score for each frame of the video input and averaging the scores over all the frames; mean results are reported in Figure 4.6.

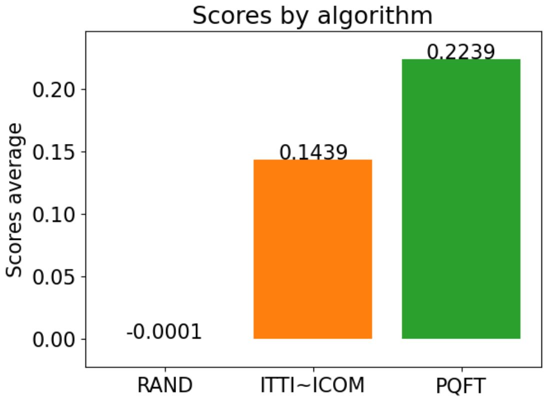

Figure 4.5: The LCC score for the three different models (ITTI and ICOM are equivalent in the spatial domain) averaged on all the images of the MIT/Tuebingen Saliency Benchmark dataset.

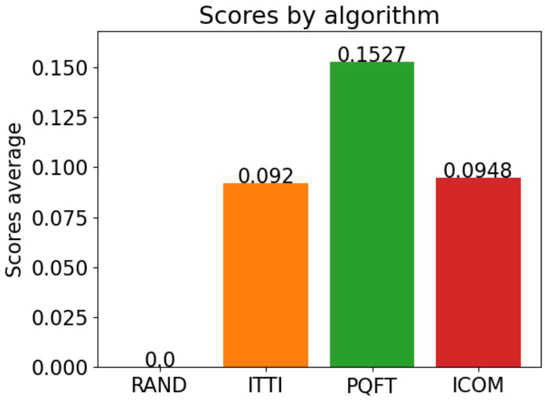

Figure 4.6: The LCC score for the four different models averaged on the first 50 videos of the dataset.

4.3.4 String edit distance

Finally, here we use also the string edit distance to compute the similarity between saliency maps. Differently from the others, this measure natively takes into account the temporal aspect.

By dividing the input image in a grid of regions it is possible to represent a sequence of fixations as the list of regions where the human eye fixates. If we then identify each region of the grid with a letter (generally a symbol), then the scanpath corresponds to a word composed by the sequence of letters (symbols) that identify the regions in the list. Analysing two different scanpaths we obtain two different words, therefore we can measure the difference between one word and the other by using edit distance measures like the Levenshtein distance or the Hamming distance.

In this case, we use each fixation map (one per frame) to get the most salient region and add its identifying symbol to the string that represents the scanpath; therefore we obtain, for each video, a string with the identifiers of each most-salient region which represents an average scanpath of the users. Similarly, for each estimated saliency map (one per frame) we get the symbol of the most salient region and append it to the string that represents the estimated scanpath.

Finally, having two strings of the same length we can compare them with the help of Hamming distance and get a value that represents the difference between the two scanpaths, namely the difference between ground truth, which is an average of the observers' scanpaths, and the estimated scanpath.

Since the videos have different durations in terms of seconds and in terms of frames, the measure of the Hamming distance here is normalized by dividing it by the length (in frames) of the video, in order to obtain a measure in the range $[0, 1]$. The lower the distance, the better the estimation resembles ground-truth saliency, a distance of 0 means that the algorithm distributes the estimated saliency always in the mostly fixated region, a distance of 1 on the contrary means that the higher concentration of estimated saliency is never in the mostly fixated region.

The size of the regions has been decided by using the standard measure of 1 degree of visual angle, which in the case of the video dataset used here is ~ 30 image pixels.

Note that this one is a measure that intrinsically takes into account the time aspect of the experiment (the time in which the scanpath is performed by the observer), therefore we use it only to evaluate algorithms on the spatiotemporal saliency estimation.

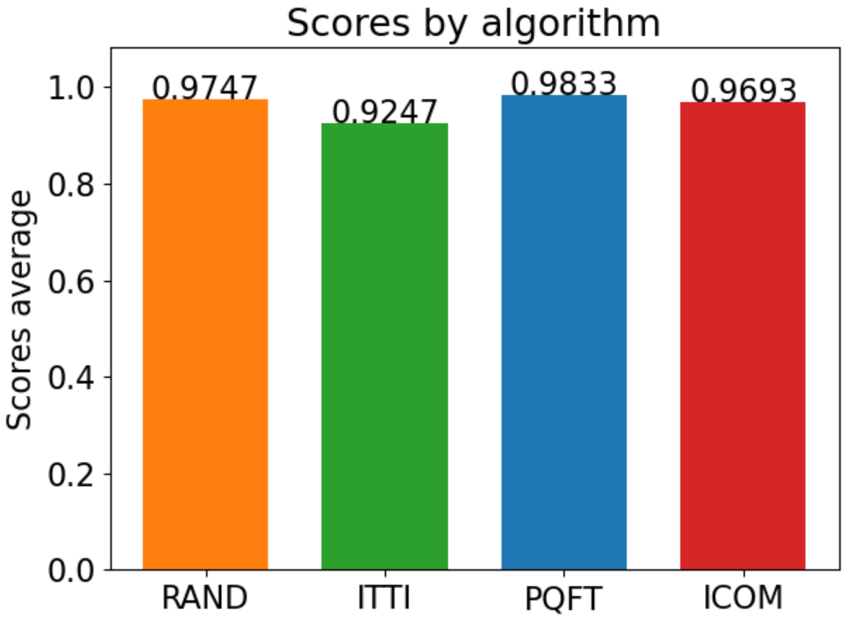

Figure 4.7: The normalized Hamming distance score for the four different models averaged on the first 50 videos of the dataset. The lower the distance value, the better the model predicts saliency.

4.4 Discussion

Having explained all the metrics and showed the results, now I propose a summary to evaluate clearly the performance of the models.

AUC

Basing on AUC score we can see that PQFT model has the highest accuracy, while ICOM model is only slightly better than ITTI model: this can be explained by a poor choice of parameters in the motion estimation component as well as particular characteristics of the videos. More precisely, some videos in the dataset have cuts and sudden changes of scene, which can cause noise in the component dedicated to motion detection.

Interestingly also the AUC score of the RAND model is pretty high, the reason of this can be found in the particular behaviour of the AUC score, which tends to ignore low-saliency false positives (see Bylinskii et al. 2019): RAND model produces maps with a lot of false positives, but their saliency level is well distributed in the range $[0, 1]$; at the same time, it is more difficult that an ESM generated by RAND contains a big number of false negatives, therefore RAND model is prone to achieving high AUC scores.

Kullback-Leibler divergence

Similar results derive from the analysis of Kullback-Leibler divergence. Both in the spatial and in the spatiotemporal domains PQFT model proves to be more accurate than the other models, ICOM scores only slightly better than ITTI and the worst model is again RAND.

Like in the case of AUC, the KL-divergence is biased to give less importance to false positives than false negatives: RAND model, which presents lots of false positives, manages to obtain a good score, even if the number of false positives is high.

Linear correlation coefficient

That is not the case for linear correlation coefficient, which clearly shows the differences in accuracy between RAND and the other models. PQFT obtains better performance than the other models, and all the models demonstrate to better estimate saliency in the spatial than in the spatiotemporal domain. This last observation reinforces the hypothesis that the motion feature is penalizing all the models, therefore the video stimuli may present non-natural motion behaviours, but a further analysis in this direction has not produced results: excluding the videos with non-natural cuts didn't lead to substantial differences in the scores; further research on this matter has to be performed in the future.

String edit distance

Finally, the normalized Hamming distance shows that it is a hard task to exactly predict where the human will fixate his gaze in the visual field since all the scores are close to 1. In this case, the model that better predicts the position of the human eye is ITTI, immediately followed by ICOM, where PQFT performs slightly worse than randomly guessing the position. This is explained by the tendency of the Hamming distance measure to give an advantage to algorithms that produce higher levels of saliency overall in the estimated saliency maps.

4.5 Consideration on processing time

One important aspect while dealing with image processing is the complexity of the algorithms and, therefore, the execution time of the algorithms. In developing ICOM model I put some effort into creating an efficient implementation of the algorithm.

The reason behind this being that the scope of this algorithm is to be part of a software that simulates shifts of attention in visual perception.

In order to create an efficient implementation of the model, I decided to exploit the characteristic of the model to compute conspicuity maps for different features in an independent way: this means that each conspicuity map can be created by an independent process. The natural continuation of this observation is that it's possible to use machines with multiprocessing capabilities to enhance the algorithm's performance.

To test what's the actual advantage in the use of this strategy I implemented a first version of the algorithm which used only one core to produce saliency maps, and then a second version which computes each feature conspicuity map in a different process. The result is a reduction in the average processing time of $\sim 31\%$, passing from a processing time of 2.01s per image to one of 1.31s.

This reduction is pretty noticeable, also because the original implementation of ITTI model, that estimates spatial saliency and therefore has to deal with one less feature, has an average execution time of 1.64s: the parallelized version of ICOM adds a functionality (motion detection) but reduces processing time. Furthermore, if we created a parallelized version of ITTI then it would still be only slightly better than ICOM because the heaviest (in a computational sense) feature to deal with is colour, which is present both in ITTI and ICOM and ultimately defines the lower limit to the processing time in both of the models.

Finally, to give a clear summary of the processing times of the various models, I report in Table 4.1 the processing times for each model (except RAND) measured as the average times registered during the evaluation phase on the video dataset. It's clear from the data reported that PQFT has by far the best performance between the models considered here. Anyway, since the scope of the model is to be used to simulate saccade movements, which are performed on average 3/4 times per second, using ICOM model could be feasible, if the processing time is reduced even only by four or five times.

model	mean time (s)	standard deviation (s)
ITTI	1.638	0.360
PQFT	0.035	0.006
ICOM	1.309	0.261

Table 4.1: Here are reported the mean values and standard deviations of processing times for the three models. PQFT is by far the quickest model. ITTI model has been evaluated in a single core evaluation, and it results slower than the parallelized version of ICOM model.

5 Conclusion

5.1 Summary

In this work I have introduced the problem of visual object differentiation and discussed a possible solution based on the implementation of an algorithm that simulates saccadic movements of the human eyes.

To simulate saccades a fundamental component is some software for saliency estimation, therefore here I proposed a new model of saliency estimation: ICOM. It is an extension of the previous ITTI model that adds a component for motion detection (Diamond Search motion detection) that enables the model to perform better in the case of spatiotemporal saliency detection.

I evaluated the performances of the model proposed comparing it to two other state-of-the-art models and a baseline. The model acquires accuracy scores comparable to some important models in the literature, but is still not an improvement. In addition, the processing time of the model is still too large for real-time implementation.

ICOM has been developed in such a way to be easily merged into a future model of human visual attention: the various methods have been implemented following the philosophy of modularity (hence the choice of different independent feature channels) so that it will be simple to introduce a *top-down component* to strengthen the attention on particular influences. This property turns out to be particularly important in cases of visual search, where it has been shown knowing that some objects have particular features (i.e. a particular colour) can be used by top-down influences to guide eye movements (see Section 2.3.2).

The model also presents the useful property of being highly parallelizable. The positive impact of the multiprocessing approach has been explored and confirmed in the evaluation phase.

The contributions that this document describes are:

- a new approach to the problem of the segmentation of a video stream in different visual objects;
- ICOM model with a Python implementation[1];
- a new python implementation of ITTI model[2];
- a new python implementation of PQFT model[3];
- an evaluation of ICOM model against ITTI, PQFT and a random control model on two different datasets of the literature.

5.2 Future developments

Clearly, the potential of the motion detection component can still be improved, this is the first of the future developments of this work. The processing time aspect is another problem that must be addressed to get performances suitable for real-time use of the saliency estimation model. Finally, the creation of a component for handling top-down influences on saliency estimation is a necessary step to build a model of human visual attention.

[1] ICOM model source code: https://github.com/Samaretas/ICOM-saliency-detection

[2] ITTI model source code: https://github.com/Samaretas/ITTI-saliency-detection

[3] PQFT model source code: https://github.com/Samaretas/PQFT-saliency-detection

Bibliography

Bensmaia, S. J. et al. (2008). “The representation of stimulus orientation in the early stages of somatosensory processing.” In: *The Journal of neuroscience : the official journal of the Society for Neuroscience* 28(3), pp. 776–786. DOI: 10.1523/JNEUROSCI.4162-07.2008.

Borji, Ali and Itti, Laurent (2013). “State-of-the-Art in Visual Attention Modeling”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.1, pp. 185–207. DOI: 10.1109/TPAMI.2012.89.

— (May 2015). “CAT2000: A Large Scale Fixation Dataset for Boosting Saliency Research”. In.

Bylinskii, Zoya et al. (2019). “What Do Different Evaluation Metrics Tell Us About Saliency Models?” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41.3, pp. 740–757. DOI: 10.1109/TPAMI.2018.2815601.

Cannon, Mark W. and Fullenkamp, Steven C. (1996). “A model for inhibitory lateral interaction effects in perceived contrast”. In: *Vision Research* 36.8, pp. 1115–1125. ISSN: 0042-6989. DOI: https://doi.org/10.1016/0042-6989(95)00180-8. URL: https://www.sciencedirect.com/science/article/pii/0042698995001808.

Engbert, R. et al. (2015). “Spatial statistics and attentional dynamics in scene viewing.” In: *Journal of Vision* 15(1).14, pp. 1–17. DOI: 10.1167/15.1.14.

Erculiani, Luca, Giunchiglia, Fausto, and Passerini, Andrea (2020). *Continual egocentric object recognition.* arXiv: 1912.05029 [cs.CV].

Findlay, J. M. (1997). “Saccade Target Selection During Visual Search”. In: *Vision Research* 37.5, pp. 617–631. ISSN: 0042-6989. DOI: https://doi.org/10.1016/S0042-6989(96)00218-0. URL: https://www.sciencedirect.com/science/article/pii/S0042698996002180.

Findlay, J.M. and Gilchrist, I.D. (Dec. 2012). “Visual attention - A fresh look”. In: *Psychologist* 25, pp. 900–902.

Guo, Chenlei, Ma, Qi, and Zhang, Liming (2008). “Spatio-temporal Saliency detection using phase spectrum of quaternion fourier transform”. In: *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8. DOI: 10.1109/CVPR.2008.4587715.

Hayhoe, M. and Ballard, D. (2005). “Eye movements in natural behavior.” In: *Trends in cognitive sciences* 9.4, pp. 188–194. DOI: 10.1016/j.tics.2005.02.009.

Huang, Yu-Wen et al. (Mar. 2006). “Survey on Block Matching Motion Estimation Algorithms and Architectures with New Results”. In: *VLSI Signal Processing* 42, pp. 297–320. DOI: 10.1007/s11265-006-4190-4.

Hurlbert, Anya (May 2003). “Colour Vision: Primary Visual Cortex Shows Its Influence”. In: *Current biology : CB* 13, R270–2. DOI: 10.1016/S0960-9822(03)00198-2.

Itti, Laurent, Dhavale, Nitin, and Pighin, Frederic (Jan. 2004). “Realistic Avatar Eye and Head Animation Using a Neurobiological Model of Visual Attention”. In: *Proceedings of SPIE - The International Society for Optical Engineering* Vol. 5200. DOI: 10.1117/12.512618.

Itti, Laurent, Koch, Christopher, and Ernst, Niebur (1998). “A model of saliency-based visual attention for rapid scene analysis”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20.11, pp. 1254–1259. DOI: 10.1109/34.730558.

Judd, Tilke, Durand, Frédo, and Torralba, Antonio (2012). “A Benchmark of Computational Models of Saliency to Predict Human Fixations”. In: *MIT Technical Report.*

Kaspar, Kai and König, Peter (July 2011). “Overt Attention and Context Factors: The Impact of Repeated Presentations, Image Type, and Individual Motivation”. In: *PloS one* 6, e21719. DOI: 10.1371/journal.pone.0021719.

Koch, Kristin et al. (Aug. 2006). “How Much the Eye Tells the Brain”. In: *Current biology : CB* 16, pp. 1428–34. DOI: 10.1016/j.cub.2006.05.056.

Land, M. and McLeod, Peter (Jan. 2001). “From Eye Movements to Actions: How Batsmen Hit the Ball”. In: *Nature neuroscience* 3, pp. 1340–5. DOI: 10.1038/81887.

Le Meur, O. et al. (2006). “A coherent computational approach to model bottom-up visual attention”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.5, pp. 802–817. DOI: 10.1109/TPAMI.2006.86.

MacInnes, W. and Klein, Raymond (Apr. 2003). “Inhibition of Return Biases Orienting During the Search of Complex Scenes”. In: *TheScientificWorldJournal* 3, pp. 75–86. DOI: 10.1100/tsw.2003.03.

Millikan, Ruth Garrett (2000). *On Clear and Confused Ideas: An Essay about Substance Concepts.* Cambridge Studies in Philosophy. Cambridge University Press. DOI: 10.1017/CBO9780511613296.

Posner, Michael (Mar. 1980). “Orienting of Attention”. In: *The Quarterly journal of experimental psychology* 32, pp. 3–25. DOI: 10.1080/00335558008248231.

Rosenholtz, Ruth, Huang, Jie, and Ehinger, Krista (Feb. 2012). “Rethinking the Role of Top-Down Attention in Vision: Effects Attributable to a Lossy Representation in Peripheral Vision”. In: *Frontiers in psychology* 3, p. 13. DOI: 10.3389/fpsyg.2012.00013.

Schwetlick, Lisa et al. (Apr. 2020). *Modeling the effects of perisaccadic attention on gaze statistics during scene viewing.* DOI: 10.31234/osf.io/zcbny. URL: psyarxiv.com/zcbny.

Shan, Zhu and Kai-Kuang, Ma (2000). “A new diamond search algorithm for fast block-matching motion estimation”. In: *IEEE Transactions on Image Processing* 9.2, pp. 287–290. DOI: 10.1109/83.821744.

Treisman, A. M. and Gelade, G. (1980). “A feature-integration theory of attention.” In: *Cogn Psychol* 12(1).14, pp. 97–136. DOI: 10.1016/0010-0285(80)90005-5.

Trukenbrod, H. A. et al. (2019). “Spatial statistics for gaze patterns in scene viewing: Effects of repeated viewing.” In: *Journal of Vision* 19(6).5, pp. 1–19. DOI: 10.1167/19.6.5.

Wang, Wenguan, Shen, Jianbing, Guo, Fang, et al. (2018). “Revisiting Video Saliency: A Large-scale Benchmark and a New Model”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.*

Wang, Wenguan, Shen, Jianbing, Xie, Jianwen, et al. (2019). “Revisiting Video Saliency Prediction in the Deep Learning Era”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence.*

Wolfe, J. and Horowitz, T. (2017). “Five factors that guide attention in visual search”. In: *Nature Human Behaviour* 1.

Yarbus, Al’fred Luk’yanovich (1967). *Eye Movements and Vision.* New York: Plenum Press.