A Neuromorphic Proto-Object Based Dynamic Visual Saliency Model with a Hybrid FPGA Implementation

Jamal Lottier Molin, *Member, IEEE,* Chetan Singh Thakur, *Senior Member, IEEE,* Ernst Niebur, Ralph Etienne-Cummings, *Fellow, IEEE,*

***Abstract*—Computing and attending to salient regions of a visual scene is an innate and necessary preprocessing step for both biological and engineered systems performing high-level visual tasks including object detection, tracking, and classification. Computational bandwidth and speed are improved by preferentially devoting computational resources to salient regions of the visual field. The human brain computes saliency effortlessly, but modeling this task in engineered systems is challenging. We first present a neuromorphic dynamic saliency model, which is bottom-up, feed-forward, and based on the notion of proto-objects with neurophysiological spatio-temporal features requiring no training. Our neuromorphic model outperforms state-of-the-art dynamic visual saliency models in predicting human eye fixations (i.e., ground truth saliency). Secondly, we present a hybrid FPGA implementation of the model for real-time applications, capable of processing 112×84 resolution frames at 18.71 Hz running at a 100 MHz clock rate — a $23.77\times$ speedup from the software implementation. Additionally, our fixed-point model of the FPGA implementation yields comparable results to the software implementation.**

***Index Terms*—Saliency, Dynamic, Motion, FPGA, Real-time, Proto-object**

I. Introduction

The ability to attend to salient regions of a visual scene is a critical preprocessing step for both biological and engineered systems. In the human visual system (HVS), each optic nerve transmits input from retinal ganglion cells to the brain in the form of spikes, otherwise referred to as action potentials [1]. The rate at which these cells transmit neural information is equivalent to the brain receiving ~ 100 Mbps of spatial and temporal visual input per eye [2]. Processing this overwhelming amount of data in parallel, and in real-time, is impossible for any human brain. To overcome this complexity, the HVS instead utilizes selective attention and attends preferentially to the most salient regions of the visual field. Psychophysical and neurophysiological evidence supports the existence of a retinotopic saliency map computed within the HVS [3]–[6].

J. L. Molin is with Northrop Grumman, 1580 W Nursery Road, Linthicum Heights, MD 21090 USA e-mail: jamal.molin@ngc.com. C. S. Thakur is with the Department of Electronic Systems Engineering, Indian Institute of Science, Bangalore-560012 India e-mail: csthakur@iisc.ac.in. E. Niebur is with the Department of Neuroscience, Johns Hopkins University, Baltimore, MD 21218 USA e-mail: niebur@jhu.edu. R. Etienne-Cummings is with the Department of Electrical and Computer Engineering, Johns Hopkins University, Baltimore, MD, 21218 USA e-mail: retienne@jhu.edu

Manuscript received July 10, 2020; revised Nov 22, 2020.

Visual saliency also serves as an aid in the field of computer vision. It is a vital initial processing step for engineered systems designed to perform high-level visual tasks in real-time including, but not limited to, navigation and localization, object detection, classification, tracking, image/video compression, surveillance and security, and action recognition. It enables data triaging by detecting and transmitting only relevant data to subsequent layers of processing. This frees up computational resources required for high-level tasks and reduces data dimensionality and processing latency [7]–[9].

We present a novel neuromorphic model of dynamic visual saliency, which takes into account the temporal dynamics of the scene, inspired by the HVS. By doing so, we further bridge the gap between biological systems and engineered systems, resulting in an improved model of dynamic visual saliency for predicting human eye fixations.

There are two components of visual attention, bottom-up and top-down. Top-down attention is a function of the viewer's biases based on their internal state and goals and is not considered in this study. We focus instead on bottom-up attention, which is a function only of the inherent properties of the visual stimulus itself and often referred to as saliency. An early, purely functional (not computational) model of visual saliency by Koch and Ullman [10] was feature-based. This lead to the development of arguably the most influential early model of feature-based visual attention by Itti, Koch and Niebur [11].

Going beyond feature-based models, more recent object-based saliency models are supported by Gestalt pyschology. They are based on the idea that scenes are represented and manipulated not in terms of elementary features, but in terms of perceptual objects — both static and dynamic [12]–[20]. One hypothesis explaining object-based attention is Rensink's coherence theory [21]. It states that elementary features are organized in low-level "proto-objects", which are formed rapidly and in parallel across the visual field. These proto-objects are pre-attentive structures with limited spatial and temporal coherence. Focused attention is required to stabilize them, which is possible only for a small number of proto-objects. This generates the perception of an object with a much higher degree of coherence over space and time (these ideas were foreshadowed by Treisman's Feature Integration Theory of Attention [22]). Once attention is released, the object dissolves back to its dynamic proto-object state [21].

In this study, we develop a proto-object based, bottom-up,

computational saliency model based on the intrinsic properties of visual stimuli and the neuronal dynamics at early stages of HVS processing. In addition to static features, motion is an important signal for the computation of saliency, both in biology and in our model. In prior, preliminary work [23], [24], we investigated multiple methods for integrating motion, utilizing a smaller video dataset. However, in this work, we have determined a more accurate and biologically plausible method for integrating temporal features for computing dynamic visual saliency, which considers motion exhibited within the scene. Additionally, we validate the model by utilizing a complete video dataset. Finally, in this work, we describe a novel hybrid FPGA implementation for real-time applications.

The structure of our model is based on the proto-object based model by Russell et al. [14] – a bottom-up, feed-forward computational model of visual saliency that computes saliency as a function of figure-ground relationships attained via the notion of proto-objects. Although biologically plausible and capable of predicting human eye fixations on static scenes better than other state-of-the-art (SOTA) saliency models, the Russell et al. model did not take into account the temporal characteristics of visual stimuli. Taking this into account is critical, given motion is the most significant contributor when computing visual saliency [25].

An additional contribution of the present study is that it goes beyond previous work in which visual saliency was computed in software. Except in very simple cases, software models do not allow for real-time, real-world applications. Therefore, we present a hybrid Field Programmable Gate Array (FPGA) implementation of the model, which offloads the most computationally expensive components of the model onto an FPGA to speed up processing. The main innovative contributions of this work are thus as follows:

1) We present a neuromorphic, dynamic visual saliency model, which takes into account motion in the visual scene. Motion is integrated into the proto-object based visual saliency model in a biologically plausible manner that is sufficient for computing bottom-up visual saliency on videos (i.e., dynamic scenes). This model is based solely on neurophysiological properties of the HVS, thus parameters are fixed and no training is required. This is an immense advantage over conventional machine learning based models, which require computationally demanding training procedures and suffer in the case of data unavailability. Training deep neural networks, with hundreds of thousands of parameters is also very costly, in terms of time, energy and other resources, and may soon be constrained by very severe economical and ecological limitations [26]. In addition, these models are subject to problems due to overfitting and lack of generalization, and they are sensitive to spurious inputs, as in adversarial attacks [27]. None of these constraints apply to our model, and, in addition, it outperforms other SOTA dynamic saliency models in computing saliency such that human eye fixations are considered ground truth.
2) We introduce a novel hybrid hardware implementation of this dynamic proto-object based visual saliency model on an FPGA, making possible the real-time processing that we seek. While inexpensive computation is performed in software, the computationally expensive components of the model are implemented on an FPGA to speed up processing, taking us closer to an end-to-end FPGA implementation. The system is capable of predicting human eye fixations on both static and dynamic visual stimuli, at frame rates that are acceptable for real world settings.

II. Related Work

A. SOTA Dynamic Visual Saliency Models

Many current models of visual saliency, both object-based and feature-based, compute saliency only on static visual stimuli and do not consider motion that may exist within the visual scene. To validate these saliency models, datasets of static images with corresponding human eye fixation data are used to quantify the extent to which the saliency model predicts eye fixations. However, the world is dynamic and constantly changing. Motion is a naturally occurring phenomenon that plays an important role in both human and computer visual processing, and specifically, in visual attention. For human observers, it has been shown that given a dynamic visual stimulus, motion plays a more significant role in visual saliency than other low-level features [25]. Thus, it is important to consider the temporal dynamics of the visual stimuli when computing visual saliency. More recently, saliency models have been implemented that do consider motion when computing a dynamic saliency map.

Rosenholtz [28] introduced a dynamic saliency model, which interprets saliency as an outlier to a statistical distribution of motion features. Similarly, Gao et al. [29] developed a model, which considers motion using biologically plausible spatio-temporal Gabor filters and computes the Kullback Leibler divergence between distributions of pixel feature responses from the pixel's local region. Seo et al. [30] proposed a self-resemblance method for computing saliency using statistics to measure likelihood of saliency at a given pixel relative to its local neighborhood. Itti and Baldi [31], [32] introduced a saliency model, which considers motion as an additional feature using Bayesian surprise. This model is also based on statistics and uses Bayes' theorem to statistically compute how much a new observation differs from its prior. Zhang et al. [33] computed saliency on dynamic scenes by computing dynamic saliency based on statistics and requires learning the probability distribution for each spatio-temporal feature. Fang et al. [34] proposed a novel algorithm, which computed saliency on video based on both spatial and temporal cues using an adaptive entropy-based uncertainty weighting approach.

Itti et al. [32] extended their original model to include two additional feature channels: flicker (onset/offset) and motion channels (the onset/offset channel was already included in an earlier model by Niebur and Koch [35]). Harel et al. [36] reformulated this Itti et al. method from a graph-based perspective.

In its original state, this model did not consider motion. However, its software implementation included the option for a motion channel [37]. Nonetheless, this graph-based approach is less biologically plausible in its computation. Other feature-based dynamic saliency models which consider motion include [38]–[42].

More recent feature-based models are built on machine learning-based designed to take into account compute dynamic saliency [43]–[47]. However, these learning methods require computationally expensive and time-consuming training steps (e.g. backpropagation), large amounts of training data [46], and may not be biologically plausible in their computations.

Our neuromorphic, bottom-up, object-based, dynamic visual saliency model is based on the notion of proto-objects using spatial and temporal filters analogous to neuronal receptive fields of visual processing. There is no training data nor learning required for this model as it utilizes parameters extracted from neurophysiological evidence. It does so by incorporating temporal filters based on receptive fields derived from simple cell activity observed in visual cortex V1 in the parvoceullar and magnocellular pathways of the HVS. Therefore, no computationally expensive parameter learning is required, making this model immediately applicable to new environments.

B. SOTA FPGA-based, Real-time Saliency Models

Considering the computational complexity of this model, we accelerate the computation of the dynamic visual saliency map using a hybrid FPGA implementation. This allows for real-time processing of a reliable, biologically plausible dynamic visual saliency model that is capable of predicting human eye fixations better than other SOTA models.

All of the visual saliency models previously discussed were implemented in software and run on CPUs for proof-of-concept. More recently, there has been increasing interest in implementing visual saliency models on FPGAs for real-time, low-power processing. Bouganis et al. [48] accelerated a saliency model proposed by Li et al. [49], which operated on the gray-scale of a single image only and utilized neuron models tuned to specific orientation and spatial locations. The differential equations used to model these neurons for computing saliency are computationally demanding, and therefore, the array of neurons with their associated dynamics was implemented on FPGA. Using the parallel architecture of the FPGA demonstrated a speedup of more than $10\times$.

Kestur et al. [50] utilized FPGA to implement a library for saliency computation based on the Itti et al. (1998) model [11]. This FPGA-based accelerator is called Streaming Hardware Accelerator with Run-time Configurability (SHARC) and showed $5\times$ speedup compared to the software version of the saliency model on 256×256 images. Akselrod et al. [51] utilized their NeuFlow platform of implementing a simplified Itti et al. saliency model showing a $4\times$ speedup on 480×480 images in comparison to a software implementation. Motion was also incorporated into the model. Kim et al. [52] also implemented the model on FPGA, simplifying the normalization operation for FPGA. Their implementation interfaces with a silicon retina chip and extracts various features on 128×128 images. They were able to show a speedup of more than $2.5\times$ and power reduction of more than $32\times$ by using the FPGA implementation. Moradhasel et al. [53] designed an FPGA based saliency model and showed computation speeds of 50 million pixels per second. Similarly to the Akselrod et al. [51] implementation, they also considered motion in this model. It showed a $2\times$ speedup over SOTA models at that time. Most recently, Barranco et al. [54] developed a simplified, yet more complete FPGA implementation of the saliency model incorporating motion as well as winner-take-all and inhibition of return. It also has a top-down component, which modulates the final saliency map as a function of optical flow and depth. This model outperformed all previous models with respect to speed as it computed saliency maps at 180 fps for 640×480 resolution. Finally, other saliency models have been implemented on FPGA including the work of Bae et al. [55] where the AIM (Attention based on Information Maximization) algorithm by Bruce et al. [56] was implemented on an FPGA platform for real-time processing capable of 4 million pixels per second.

The FPGA implementations of saliency models discussed demonstrate the advantages that FPGA implementations have over purely software implementations with regard to processing speed. However, all of these models implemented on hardware are purely feature-based and do not compute saliency as a function of objects. The hybrid FPGA implementation presented here is, to our knowledge, the first proto-object based model running in real-time by utilizing a low size, weight, and power (low SWaP) hardware accelerator platform. This brings us closer to an end-to-end low SWaP proto-object based dynamic saliency model for real-world applications. By implementing our proto-object based model in hardware, we achieve speedup in processing as well as accurate saliency computation.

III. Our Neuromorphic Model of Dynamic Saliency

We name our neuromorphic model of dynamic visual saliency PODVS (Proto-Object Based Dynamic Visual Saliency). It utilizes the idea of separable space-time filters for incorporating motion. This biologically plausible model is based on the idea that simple cells in the magnocellular and parvocellular pathways act as spatio-temporal filters. They not only extract spatial information preattentively, but also temporal information. Spatially, information is extracted within grayscale and color-opponency versions of the incoming frame, as well as edge and center-surround spatial filtering. Strongly phasic and weakly phasic temporal filtering is applied within the various channels in order to extract dynamic features. This model utilizes the ideas of the original proto-object based visual saliency model by Russell et al. [14], however, now extracts both temporal and spatial information for computing saliency on dynamic visual stimuli. This allows for computing saliency in videos by considering motion that may exist within the scene. A block diagram of the model can be seen in Figure 2. The code is also publicly available [57].

A. Proto-Object Based Saliency Model for Static Images

Our model is inspired by the original model of proto-object based saliency by Russell et al. [14] for static images. Therefore, it is important to discuss this original model prior to discussing our saliency model for dynamic visual stimuli (i.e. video). The Russell et al. model is an object-based, bottom-up, feed-forward model of visual saliency. It is based on the notion of proto-objects, which may exist within the visual field. The model outperformed other SOTA models [11], [36] of visual saliency on predicting human eye fixations on a dataset of static images of natural scenes. The model works as follows: It receives an RGB (Red, Green, Blue) color image (resolution of 640×480) and decomposes this image into three feature channels: intensity, color, and orientation. Within each of these feature channels are sub-channels. The intensity channel has one sub-channel. The color channel has four sub-channels: red-green opponency, green-red opponency, blue-yellow opponency, and yellow-blue opponency. The orientation channel also has four sub-channels (four orientations): $0°$, $45°$, $90°$, and $135°$. This results in a total of 9 feature channels. Once the original color image is decomposed into these 9 channels, within each channel, the feature map is successively down-sampled in steps of $\sqrt{2}$ to form an image pyramid spanning five octaves. Proto-object activity is then computed within each channel and at each level of the pyramid, independently. Proto-object activity gives rise to saliency with respect to figure-ground relationship within the visual scene. The proto-objects are computed using a grouping mechanism consisting of edge and center-surround operators working together to compute border ownership activity. Neurons encoding border ownership (one-sided assignment of a border to a region perceived as a figure) have been discovered in early stages of visual processing, predominantly in visual cortex V2, by Zhou et al. [58]. This border ownership activity is integrated in a circular fashion to reveal grouping activity. More details on this grouping mechanism can be found in Ref. [14]. A normalization operation, N_1, is then applied to each grouping activity map to enhance maps with single proto-objects and suppress maps with multiple proto-objects. This normalization operator, N_1, works as follows:

1) The maximum, m, of the map being normalized is determined.
2) The average of the other local maxima, $\bar{m}$, is determined.
3) Finally, there is a global, element-wise multiplication of the map by $(m - \bar{m})^2$.

This normalization, N_1, is a function of the grouping activity such that grouping activity with few proto-objects is promoted while grouping activity of maps with multiple proto-objects is suppressed. Following this normalization, a similar computation to that in the Itti et al. (1998) model [11] is performed. The image pyramids within each channel are collapsed by scaling each level to a common level and summing. This results in a single conspicuity map within each channel. These 9 conspicuity maps are then normalized using a second normalization operator, N_2. The normalization operator, N_2, works as follows:

1) The map is normalized to the range $[0, ..., M]$.
2) The maximum, m, of the map being normalized is determined.
3) The average of the other local maxima, $\bar{m}$, is determined.
4) Finally, there is a global, element-wise multiplication of the map by $(m - \bar{m})^2$.

The only difference in this normalization, N_2, is the additional first step of normalizing each map to a common range $[0, ..., M]$. This step is necessary for allowing invariance to modality (feature). This globally enhances conspicuity maps with few strong peak responses and globally suppresses maps with many comparable peak responses. Finally, these normalized conspicuity maps are linearly summed to form the final saliency map.

B. Motion Processing in the Human Visual System

We seek to represent motion in our model in a biologically plausible manner. This model is purely bottom-up and feed-forward. Henceforth, we focus on how motion is computed at early stages of visual processing, and further, preattentive visual processing. Neurophysiological research has shown that motion extraction occurs along the dorsal pathway beginning in V1 and proceeds to middle temporal area (MT) and then continues to the medial superior temporal area (MST) [59]. Motion extraction in V1 can be represented by local spatio-temporal filters and shows preference to spatial frequency, spatial phase, spatial orientation, and direction of motion. Later stages of motion are responsible for computing velocity and optical flow. This motion processing requires attention [60]. However, we are interested in representing preattentive motion, and therefore, consider motion extraction in V1 only.

It should be noted that we are concerned with the receptive fields of non-direction selective simple cells. As previously noted, there are two pathways within the visual system: the magnocellular and parvocellular pathways [59]. Each of these pathways has a select population of retinal ganglion cells, which project to the LGN (Lateral Geniculate Nucleus), and further to primary visual cortical cells. Strongly phasic simple cells exist within the magnocellular pathway, they have high temporal resolution, high contrast sensitivity and low color sensitivity. Cells in the parvocellular pathway are weakly phasic and have low contrast sensitivity and high color sensitivity but low temporal resolution. Strongly phasic cells typically have a strong excitatory phase followed by a strong inhibitory phase. Weakly phasic cells typically have a less pronounced excitatory phase followed by a weak inhibitory phase, resulting in a weaker response to motion. Approximately 20-25 percent of the population of non-direction selective simple cells in V1 are strongly phasic. The remaining are weakly phasic. The temporal filters used in the models to be discussed are modeled to fit the receptive fields of these strongly and weakly phasic cells found in the primary visual cortex [59], [61].

1) Biologically Plausible Temporal Filters: The work of Parkhurst [62] and De Valois et al. [59] was used to model the transfer function for a biologically plausible temporal filter modeling the temporal receptive field of strongly phasic and weakly phasic, non-direction selective simple cells in V1. The approximation of the transfer function of the V1 simple cell temporal receptive field can be seen in Equation 1.

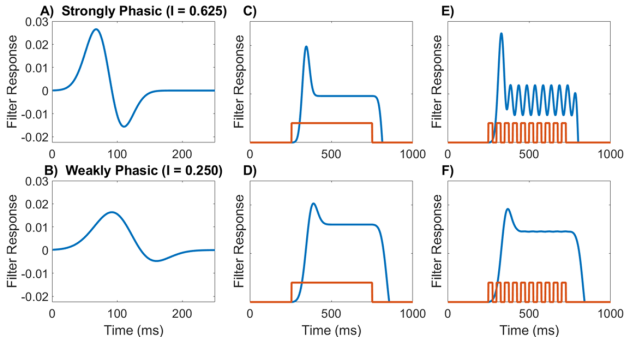


Fig. 1. Plots A and B show the temporal profile (blue line) of a strongly phasic and weakly phasic filter, respectively. Plots C, D, E, and F, show the filter response (blue line) to a stimulus (red line). Plots C and D show the filter response to an abrupt then constant stimulus. Plots E and F show the filter response to flicker motion (continuous onset/offset change). Strongly phasic filters are more sensitive to temporal change. I is the ratio of the peak positive to the peak negative amplitude of the filter, hence, representing the degree to which the filter is strongly phasic. Image derived from Ref. [62].

$$r(t) = \alpha(t - \tau - \delta)e^{\beta(t-\tau)^2} \quad (1)$$

- $r(t)$ - continuous-time filter
- α - response amplitude parameter
- β - response amplitude parameter
- τ - time shift parameter
- δ - determines degree to which weakly or strongly phasic in time

These parameters were fit to model the temporal response profile of strongly phasic and weakly phasic cells in V1 from neurophysiological recordings [62]. These parameter values can be seen in Table I.

TABLE I
PARAMETERS FOR STRONGLY/WEAKLY PHASIC V1 SIMPLE CELL TEMPORAL RESPONSE

Type	α	β	τ	δ
Strongly Phasic	-0.00161	-0.00111	86.2	5.6
Weakly Phasic	-0.000487	-0.000466	116	20

These temporal profiles for strongly and weakly phasic simple cell receptive fields are applied in our dynamic model of proto-object based visual saliency. The methodology in regards to how they are applied will be discussed in Section III-C. A visual representation of the filters can be seen in Fig. 1. The strongly phasic temporal filter in Fig. 1A, has a strong positive/excitatory lobe and a strong negative/inhibitory lobe, hence, is more sensitive to motion. The weakly phasic temporal filter in Fig. 1B, has a positive/excitatory lobe and weak negative/inhibitory lobe, and, hence, is less sensitive to motion. While the y-axis is the filter coefficient, the x-axis is time (in the past) assuming a rate of 24 frames per second for the incoming input image sequence.

Fig. 1 shows the response to various types of stimuli. The value of I is the ratio of the peak positive amplitude to the peak negative amplitude of the temporal filter, representing the degree to which the filter is strongly phasic. Hence, the strongly phasic filter has a larger value for I ($I = 0.625$) and the weakly phasic filter has a smaller value for I ($I = 0.250$). Fig. 1A and Fig. 1B are the strongly and phasic filters, respectively. Fig. 1C and Fig. 1D are their responses to an abrupt onset then constant stimulus. Fig. 1E and Fig. 1F are their responses to flicker (onset/offset) motion. The strongly phasic filter clearly has a higher response to temporal changes in the stimuli.

Because of the finite duration of the filter functions, frames more than 250 ms in the past do not contribute to saliency. Components of the visual stimulus that do not change over an extended amount of time generate a lower temporal response. Relatedly, these temporal filters have a stronger response to onset and offset of objects within the scene. These temporal dynamics are similar to those seen in the model of visual saliency depicted in Ref. [33], however, their model uses learning. In the following sections we will discuss how these temporal filters are utilized within our model, specifically, how these temporal filters are applied within each feature channel for computing proto-objects.

C. Extracting Spatio-Temporal Activity

The model receives dynamic visual stimuli (i.e. color video) as input. This input can be realized as a sequence of images (RGB video frames). We assume a resolution of 640×480 and frame rate of 24 frames per second. For each frame, a new saliency map is computed as a function of dynamic proto-objects from spatio-temporal responses within different feature channels: intensity, color, and orientation.

1) Intensity Channel: Given that simple cells in the magnocellular pathway have high contrast sensitivity, low color sensitivity, and high sensitivity to motion [59], we apply the strongly phasic temporal filter in the intensity channel. To extract the intensity of the current frame, the average of the red (r), green (g), and blue (b) channel is computed. The temporal filter is applied on the current frame and previous frames of the intensity-valued frames. The convolution is applied temporally across the video frames. The total number of frames in which the convolution is applied is dependent on the frame rate of the videos. In our case, our chosen frame rate of 24 Hz results in a filter convolution over the current frame and five previous frames. The representation of this discrete convolution can be seen in Equation 2.

$$M_S[n] = (F * R_S)[n] = \sum_{t=0}^{T} \sum_{r=1}^{N_r} \sum_{c=1}^{N_c} F_{r,c}[n-t] \times R_S[t] \quad (2)$$

$M_S[n]$ is the strongly phasic temporal output at frame n. $F_{r,c}[n]$ is the pixel intensity of the original grayscaled (intensity version) video at row r and column c at frame n. $R_S[t]$ is the discretized representation of the filter $r(t)$ in Equation 1 using the strongly phasic parameters in Table I. Finally, $F[n-t]$ represents the frame at t frames in the past. T is the total number of frames in the past over which to perform the convolution. In our case, $T = 6$. N_r and N_c are the number of rows and columns, respectively, in each frame. The output, $M_S[n]$ (strongly phasic output) is the input to the grouping stage within the intensity channel. $M_S[n]$ has dimensions $N_r \times N_c$.

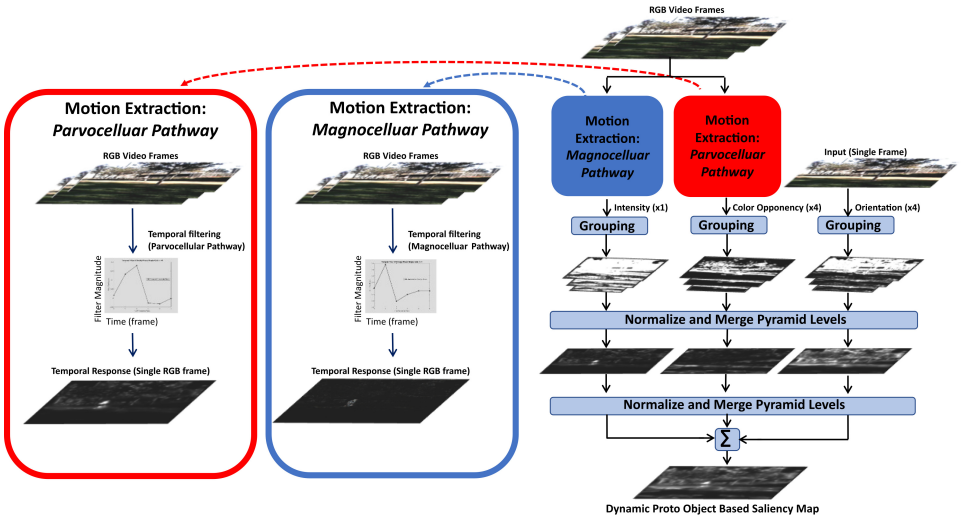


Fig. 2. PODVS - Proto-object based Dynamic Visual Saliency. This model utilizes spatial and temporal information of the dynamic scene as input to the model. The model receives RGB video frames as input. For the intensity channel, motion is extracted using strongly phasic temporal filters (magnocellular pathway). The output of this response serves as input into the grouping stage for computing dynamic, pre-attentive proto-objects in the intensity channel. In the color channels, weakly phasic temporal filters (parvocellular pathway) are used, which are less sensitive to motion and retain more static regions of the scene. The output of this spatio-temporal response serves as input to the grouping stage within the color channel. No motion is extracted within the orientation channel, therefore, static information is preserved.

2) Color Channel: Given that simple cells in the parvocellular pathway have low contrast sensitivity, high color sensitivity, and are less sensitive to motion, the weakly phasic temporal filter is applied within this channel. The same convolution is performed on the video sequence for the red, green, and blue channels. However, in this case, the discretized weakly phasic filter is applied, and henceforth, $R_W[t]$ is modeled by Equation 1 using the weakly phasic parameters in Table I. The weakly phasic motion response can be seen in Equation 3. The only difference between Equation 2 and Equation 3 is the temporal filter used.

$$M_W[n] = (F * R_W)[n] = \sum_{t=0}^{T} \sum_{r=1}^{N_r} \sum_{c=1}^{N_c} F_{r,c}[n-t] \times R_W[t] \quad (3)$$

The output, $M_W[n]$ (weakly phasic output), is the input to that of the color channel. $M_W[n]$ has dimensions $N_r \times N_c \times 3$. The RGB output after applying this weakly phasic temporal filter within each channel is used as input to the color subchannels. The color channel is made up of four subchannels. These are red-green opponency (RG), green-red opponency (GR, blue-yellow opponency (BY), and yellow-blue opponency (YB). These features are extracted by decoupling hue from intensity by normalizing each color channel by intensity. These four separable spatio-temporal filtered outputs (RG, GR, BY, YB) are used as input to the grouping stage of the color channel.

3) Orientation Channel - Spatial Content Only: Within the orientation channel, there is no temporal filtering. Extraction of temporal information within the intensity channel and color channel is sufficient. Furthermore, this helps to preserve static information with regards to saliency. In the orientation channel, there are four subchannels. Within each channel, saliency is computed in regards to salient objects with respect to a unique orientation. These four subchannels are O_0, $O_{\frac{\pi}{4}}$, $O_{\frac{\pi}{2}}$, and $O_{\frac{3\pi}{4}}$ where 0, $\frac{\pi}{4}$, $\frac{\pi}{2}$, and $\frac{3\pi}{4}$ correspond to the four unique orientations. For each of these subchannels, the grayscaled, intensity version of the current frame is the input to the grouping stage.

D. Grouping Mechanism and Normalization

The spatio-temporal responses of each of these feature channels are fed as input to the grouping stage of the model. The grouping mechanism is that used in Ref. [14]. It utilizes computational methods inspired by Craft et al. [63]. The grouping processing flow can be described in the following processing steps, P1 to P7, as listed below:

a) P1: Extract Spatio-Temporal Activity
b) P2: Generate Frame Pyramid
c) P3: Complex Edge and Center-Surround Filtering
d) P4: von Mises Filtering
e) P5: von Mises Summation
f) P6: Border Ownership Responses
g) P7: Grouping Responses

1) P1 - Extract Spatio-Temporal Activity: The spatio-temporal responses are extracted using the methods previously described. Within the intensity channgel, the strongly phasic temporal activity extracted from the grayscale frame is computed. Within the color channel, weakly phasic temporal activity of the color opponency response for each subchannel (RG, GR, YB, and BY) is computed. Finally, within each orientation subchannel, the grayscale frame is used. This initial step results in 9 derivations of the original frame, representative of the spatio-temporal activity extracted within each of the 9 subchannels.

2) P2 - Generate Frame Pyramid: To achieve scale-invariance, within each of the 9 subchannels channels, the spatio-temporal responses are successively down-sampled in steps of $\sqrt{2}$ to form an image pyramid spanning five octaves (10 pyramid levels). The grouping mechanism step is then performed independently within each subchannel. Additionally, for initial stages of the grouping mechanism, processing occurs independently within each level of each pyramid.

3) P3 - Complex Edge and Center-Surround Filtering: The first stage of this mechanism is extraction of object edges, similarly to the receptive field of simple cells in V1. Both odd and even edge responses are combined to form complex cell responses. These complex cells are contrast-invariant edge responses, which directly excite left or right side preferred border ownership neurons. Even and odd Gabor kernels of size 19×19 are used. In order to extract information regarding the existence of objects, a center surround operation is performed. This is similar to the receptive field of neurons found within the retina and LGN – both ON- and OFF-center receptive fields. This is necessary for detecting dark objects on light backgrounds as well as light objects on dark backgrounds. Difference of Gaussians kernels of size 19×19 are used to for the center-surround filtering. Each filtering operation can be seen as a weighted sum between the kernel and image patch at pixel location (x,y).

4) P4 - von Mises Filtering: To compute border ownership responses, the complex cell response is modulated by the center-surround cell responses. Excitation from the center-surround response coding for figure on the border ownership cells preferred side increases border ownership activity. Center-surround activity on the non-preferred side inhibits/suppresses border-ownership activity. To extract border ownership responses, von Mises filtering is performed using kernels of size 13×13. This operation must be performed for both left- and right-sided responses, 4 orientations, and ON- and OFF- center-surround responses resulting in 16 convolutions on each frame of each pyramid level.

5) P5 - von Mises Summation: The von Mises summation step is computed for each pyramid level of each von Mises response from step P4. This process involves, for each level k, scaling up each lower level frame s in the pyramid ($s \leq k$) and accumulating the result. This results in a new pyramid of border ownership responses for both left- and right-sided responses, 4 orientations, and light and dark object activity. These responses are defined as $B^k_{Light,L}[\theta]$, $B^k_{Light,R}[\theta]$, $B^k_{Dark,L}[\theta]$, and $B^k_{Dark,R}[\theta]$. The parameter k is the pyramid level. $Light/Dark$ signifies ON-/OFF- responses, respectively. L/R signifies left-/right-sided (θ and $\theta+\pi$) border ownernship responses, respectively. θ signifies the orientation (0, $\frac{\pi}{4}$, $\frac{\pi}{2}$, and $\frac{3\pi}{4}$).

6) P6 - Final Border Ownership Responses: Total border ownership activity for left- and right-sided borders is computed by summing across ON- and OFF-center responses. This can be seen in Equation 4 and Equation 5, respectively.

$$B^k_L[\theta] = B^k_{Light,L}[\theta] + B^k_{Dark,L}[\theta] \tag{4}$$

$$B^k_R[\theta] = B^k_{Light,R}[\theta] + B^k_{Dark,R}[\theta] \tag{5}$$

7) P7 - Grouping Responses: Finally, border ownership activity is integrated in an annular fashion to give grouping cell activity. This grouping activity is representative of dynamic proto-object activity giving rise to figure-ground relationship of the dynamic visual scene. To compute the final grouping activity, masks must be generated using a max operator on the border ownership activity that effectively computes border ownership assignment by determining the side a border belongs to – $BOMaskL^k_\theta(x,y)$ and $BOMaskR^k_\theta(x,y)$) for all pyramid levels, k, and orientations ,θ. The grouping responses for left border ownership and right border ownership are then computed independently using Equations 6 and 7, respectively.

$$\begin{aligned} GrpL^k_\theta(x,y) = BOMaskL^k_\theta \otimes B^k_L[\theta] * v_\theta \\ -w_p \times BOMaskL^k_\theta \otimes B^k_R[\theta] * v_\theta \end{aligned} \tag{6}$$

$$\begin{aligned} GrpR^k_\theta(x,y) = BOMaskR^k_\theta \otimes B^k_R[\theta] * v_{\theta+\pi} \\ -w_p \times BOMaskR^k_\theta \otimes B^k_L[\theta] * v_{\theta+\pi} \end{aligned} \tag{7}$$

The parameter w_p is the weight of the inhibitory connection to the opposing side border ownership. In this model, $w_p = 1$. Note that v_θ and $v_{\theta+\pi}$ are the von Mises kernels used for extracting for both left and right border ownership responses. The operator $\otimes$ is an element-wise operation and $*$ is the convolution operator. The final grouping step involves summing the left and right grouping responses as seen in Equation 8.

$$GrpSum^k_\theta(r,c) = GrpL^k_\theta(r,c) + GrpR^k_\theta(r,c) \tag{8}$$

This results in grouping responses, $GrpSum^k_\theta(r,c)$, for each orientation, $\theta = 0, \frac{\pi}{4}, \frac{\pi}{2}, and \frac{3\pi}{4}$, for each pyramid level, $k = 1, 2, 3, ..., N_k$), where N_k is the number of pyramid levels. These grouping responses are representative of proto-object activity.

E. Final Normalization Step

To compute conspicuity maps for each subchannel, maps across a single pyramid are normalized, rescaled to a common resolution, and summed. The same normalization techniques from Itti et al. [11] are then applied across the subchannels' conspicuity maps. The results within each channel are linearly summed to form the instantaneous saliency map at the current frame. These sequential saliency maps over time (frame by frame) form the final dynamic saliency map. This is visualized in Fig. 2.

IV. Hybrid FPGA Implementation

The grouping mechanism described for computing proto-objects is a computationally expensive task. This is due to the large number of multiply-accumulate (MAC) operations required for the various filtering operations. For this model to have real-world application and be integrated into vision systems that must process frames in real-time, a hardware realization of at least this grouping mechanism is necessary. Therefore, we have implemented a hybrid FPGA implementation

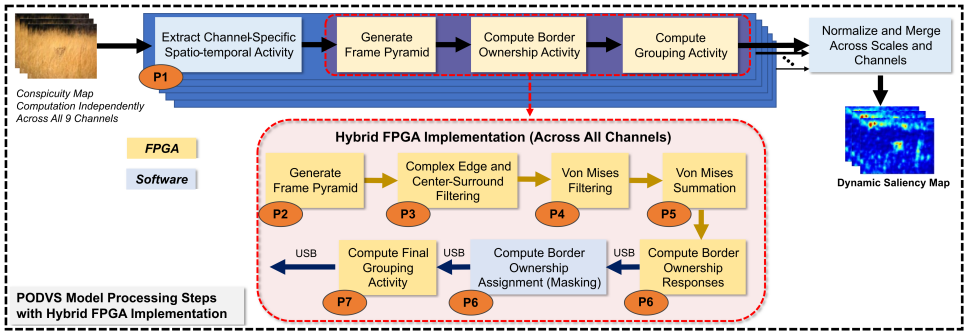


Fig. 3. Block diagram of our Proto-Object Based Dynamic Visual Saliency (PODVS) model processing flow. This also includes a visualization of the hybrid FPGA implementation signifying components of the model implemented on FPGA for speedup in processing. The yellow blocks signify the components of the model implemented on FPGA and the blue blocks signify components implemented in software. The orange circles indicate processing steps P1 to P7.

using the Opal Kelly XEM7350 Kintex-7 FPGA (XC7K160T-1FFG676C) board. The computationally demanding tasks are processed on an FPGA, while less computationally demanding tasks are processed in software on the host. We name this hybrid FPGA implementation PODVS-FPGA. In our case, the host is an Intel Quad-Core i7 CPU. The FPGA enables pipelining and parallel processing to speedup processing. It is important to note that the CPU is used for proof-of-concept, but these less computationally demanding tasks can also be implemented on the same FPGA or other local processor (e.g., ARM processor) to achieve a full end-to-end low-SWaP hardware solution. A high-level processing flow of this hybrid approach is visualized in Fig. 3. The design flow for implementation consisted of writing VHDL code for the computationally demanding processing steps. In the following sections we describe each processing component of the hybrid FPGA implementation.

A. FPGA-Host Communication

This hybrid approach incurs a latency cost for communication to and from the host and FPGA. However, the data transfer between the CPU and FPGA is minimized and communication latency is insignificant relative to the processing latency. For transmitting data, USB 3.0 is used with a bandwidth of 340 MB/s. The Opal Kelly board has a built in USB interface called FrontPanel containing an API on the host, and an HDL IP for handling the communication on the FPGA. Physical data transmission latency can be computed using Equation 9. For 112×84 resolution, physical transmission latency of a single frame is $\sim 30\ \mu s$. Even at a resolution of 640×480, the transmission latency is $\sim 1\ ms$

$$\text{Transmission Latency / Frame} = \frac{N_r \times N_c}{340 \text{ MB/s}} \tag{9}$$

B. Input Video Stream

The input to the model is a video stream. The frame resolution is of size $N_r \times N_c$, where N_r is number of rows and N_c is number of columns. In our software model, $N_r = 480$ and $N_c = 640$. While a resolution of 640×480 was ideal, the larger the resolution, more resources required. Due to limited resources, for our PODVS-FPGA implementation, $N_r = 84$ and $N_c = 112$. The bit resolution of each pixel is 8 bits. The total number of pixels to process per frame is $N_p = N_r \times N_c = 9408$ pixels.

C. P1 - Extract Spatio-Temporal Activity

Spatio-temporal activity is first extracted on the host in software. The spatial activity extraction for computing grayscale and color-opponency frames does not require any MAC operation. Additionally, the temporal responses require only 1-D convolution operation across 6 frames (opposed to 2-D filtering required for extracting spatial components). Therefore, the temporal activity response is not computationally demanding, and instead memory-demanding for storing 6 frames as input resolution increases. The objective was to utilize only internal BRAM for fast memory access. This step was not the limiting factor to achieve real-time processing, and therefore, is performed software. The combined spatio-temporal activity extraction and transfer to the FPGA incurs a latency of $\sim 30ms$. The amount of FPGA BRAM required for storing a single frame for a single channel can be calculated using Equation 10. This totals to $75.2KB$ for 112×84 resolution.

$$\text{P1 bits} = 112 \times 84 \times 8 \text{ bits} = 75264 \text{ bits} \tag{10}$$

D. P2 - Generate Frame Pyramid

The frame pyramid is generated on the FPGA using a "nearest-neighbor" downsampling method. For each new level in the pyramid, computing the pixel address in the original frame from which to subsample a pixel is approximated using bit-shifts. Computing this address takes 5 clock cycles (CC). The 112×84 is downsampled to 2 additional levels – 80×60 and 56×44. The total additional BRAM required is 57408 calculated as

$$\text{P2 bits} = \sum_{k=2}^{N_k} N_r^k \times N_c^k \times 8 \text{ bits} = 57408 \text{ bits} \tag{11}$$

where k is the pyramid level (in our case, $k = 3$), N_k is the number of pyramid levels, and N_r^k and N_c^k are the number of rows and columns of pixels at level k, respectively.

E. P3 - Complex Edge and Center-Surround Filtering

The next stage in computation involves performing parallel filtering tasks with supporting computation for extracting the complex edge response and both ON-center-surround and OFF-center-surround responses within the current feature channel. The complex edge responses are extracted by computing the square root of the even and odd edge responses. The processing steps are as follows:

1) Load 5×5 patch at the current pixel location
2) Compute weighted-sum for 8 edge and ON-center-surround response
3) Compute square root for complex edge response of all 4 orientations
4) Invert ON-center-surround response, resulting in OFF-center-surround response
5) Resulting 6 responses are stored in BRAM

For the filtering tasks, the weighted-sums are implemented in hardware as a series of MAC operations. For each of the 4 even edge kernels, 4 odd edge kernels, and ON-center-surround kernel, the MAC operations within each pyramid level frame are computed in parallel, dramatically speeding up processing. A single MAC has a latency of 5 CC (including read/write). A square root operation has a latency of 1 CC. Given parallel filtering operations (9 in parallel), the latency for this stage can be calculated as

$$\text{P3 Latency} = (\text{MAC+Sqrt CC}) \times K_r \times K_c \times (N_r^1 \times N_c^1) \quad (12)$$

where K_r and K_c are the dimensions of the kernel. The latency of the first level of the pyramid will be the longest latency given parallel operation of each filtering operation across kernels and across pyramid levels. Therefore, the latency of a single filtering operation (with the complex edge compute) on the first level of the pyramid determines the latency of step $P3$. In our case, given $K_r = 5$, $K_c = 5$, $N_r^1 = 84$, and $N_c^1 = 112$, latency for $P3$ is ~ 1.66 M CC. At the edge of each frame, MAC operations are performed by padding the edge with a pixel value equivalent to that of the adjacent edge pixel. For storing results, an additional $\sim 100KB$ of BRAM is required to store 6 pyramid responses.

F. P4 - von Mises Filtering

Von mises filtering is required for computing border ownership responses. This includes 16 parallel filtering operations per pyramid level (48 parallel convolutions for $k = 3$). The total amount of BRAM required to store all responses is ~ 265.7 KB and the total latency incurred for step $P4$ is the similar to that in Equation 12, except the square root operation is not required. This results in a latency of ~ 1.2M CC.

G. P5 - von Mises Summation

There exist 16 finite state machines (FSMs) running in parallel to compute the von Mises summation (8 for von Mises responses for light objects and 8 for von Mises responses for dark objects from previous section). Each FSM executes the following process:

1) Apply scaling factor to obtain pixel value in other pyramid level (3 CC)
2) Retrieve pixel value in bottom adjacent level (1 CC)
3) Multiply by factor 2^{-j} where j is the pyramid level (1 CC)
4) Accumulate result for current pyramid level (1 CC)
5) Repeat prior steps for each lower pyramid level, k such that $\rightarrow k \leq j$
6) Repeat all previous steps for each pyramid level j
7) Store results in BRAM used for von Mises filtering (1 CC)

This process must occur for each pixel location in each image. No additional block RAM is required as the results of the von Mises sum replace the values in the von Mises filtering responses as they are no longer required for the remainder of the model. The latency incurred for step $P5$ is $\sim$ 241K CC given $k = 3$ and resolution of 84×112.

H. P6 - Final Border Ownership Responses

The responses $B^k_{Light,L}[\theta]$, $B^k_{Light,R}[\theta]$, $B^k_{Dark,L}[\theta]$, and $B^k_{Dark,R}[\theta]$ are computed using the von Mises summation responses from step $P5$. The variable k corresponds to the pyramid level. There are 8 FSMs used for each pyramid level for computing the 2 border ownership responses (for left border ownership and right border ownership) at each of the 4 edge orientations, θ (0, $\frac{\pi}{4}$, $\frac{\pi}{2}$, and $\frac{3\pi}{4}$), and each of the 3 pyramid levels, in parallel. Therefore, 12 left and right border ownership responses are computed in parallel. This totals to 24 border ownership responses, and therefore, 24 additional instantiated BRAMs for storing the responses. Computing the left and right border ownership (4 edge orientations) for a single pixel requires 6 CC. Therefore, given a resolution of 112×84, a latency of $\sim$ 56K CC is incurred for step $P6$ and total additional BRAM required is ~ 133.3 KB of BRAM.

I. P7 - Grouping Responses

To compute the final grouping activity, masks must be generated using a local argmax operator on the border ownership activity that effectively determines the appropriate objects to which the edges (i.e. borders) belong to. To compute these masks, the border ownership responses are transmitted to the PC (host) and the masks are computed on the host and then transmitted to the FPGA. After these binary masks are received by the FPGA ($BOMaskL^k_\theta(x, y)$ and $BOMaskR^k_\theta(x, y)$) for all pyramid levels, k, and orientations, θ, the grouping responses for left border ownership and right border ownership are computed independently, and in parallel, using Equations 6 and 7, respectively. The latency is determined by the convolution operator at the first pyramid level. These two responses are then summed as seen in Equation 8. The latency incurred for processing on the FPGA is 1.9M CC and total additional BRAM required for data transmission and to store the grouping activity is $\sim$ 200 KB. Latency incurred in software for computing the border ownership masks is ~ 1 ms for a resolution of 112×84 and $k = 3$ pyramid levels. The data transfer rate between the host and FPGA totals ~ 1.3 ms.

J. Final Normalization and Saliency Map Generation

The final normalization, rescaling, and merging across the frame pyramid occurs on the host resulting in a conspicuity map for the channel being processed. The conspicuity maps computed within each of the 9 channels or normalized and merged, resulting in the saliency map at the current frame. The latency for this final step involves data transfer latency and normalization and merging latency. Transferring the final grouping results for each pyramid level and all 4 orientations to the host incurs a latency of ~ 0.6 ms. The latency for the normalization and merging operation within a single channel in software (on our CPU) can be estimated as

$$\text{Final Latency} \approx N_r^1 \times N_c^1 \times 4.37 \times 10^{-8} + 21.59 \times 10^{-3} \quad (13)$$

, which results in $\sim$ 22 ms given a resolution of 112×80 and ~ 35 ms given a resolution of 640×480.

K. Parallel Channel Computation

For the fastest processing speed, computing grouping activity within all feature channels in parallel on the FPGA is ideal. For example, implementing all 9 channel processing on the FPGA in parallel will result in a $\sim \times 9$ speedup for grouping activity processing. The FPGA used in this work had sufficient resources for computing a single channel for 112×84 video frames. However, to verify this parallel processing, input video frames of resolution 80×60 were used and the FPGA had sufficient resources for processing 3 channels in parallel. Given that this is a hybrid FPGA implementation, the host parallel processing is limited to the parallel capabilities of the host (in our case, the number of CPU cores).

V. Experimental Setup

First, we validate this novel dynamic visual saliency model (PODVS) by quantifying the model's ability to predict human eye fixations on videos. These results are further compared with other SOTA bottom-up, dynamic visual saliency models. Secondly, we validate PODVS-FPGA implementation by also quantifying the the model's ability to predict human eye fixations. Given the PODVS-FPGA implementation receives a low resolution input, for fair comparison, we validate that the FPGA implementation output is comparable to the software implementation output by using an emulation (PODVS-FIXED) of the FPGA model, which can ingest any input resolution.

A. Dataset

The dataset used for this work is the CRCNS (Collaborative Research in Computational Neuroscience) dataset created by Itti et al. [64]. It contains a set of 50 videos with 8 subjects' eye fixation data. The subjects' fixation locations were used to validate the saliency maps computed. This dataset was used both for validating the PODVS model's ability to predict human eye fixations and compared with that of other SOTA models, as well as to validate the similarity between the FPGA implementation of the model and the analogous software implementation. The smaller version of this dataset, called "MTV", was not used. More details on this dataset can be found in Ref. [64].

B. Metrics - Comparing with SOTA Models

To validate the saliency models' output, various saliency metrics can be used [65]. Here, we use two commonly used metrics for evaluating a saliency maps' ability to predict human eye fixations. The first is the area under the Receiver Operating Characteristic curve (AUC-ROC) [66]. The second is the Kullback Leibler Divergence (KLD) [67], [68]. In their original state, these metrics are sensitive to edge effects due to the filtering operations of the algorithms [33]. Therefore, these saliency models may introduce a center bias into the algorithm. Furthermore, typically, the center of the video is naturally the viewer's focus of attention, even more so at the beginning of a video [69]–[71]. To compensate for these center bias effects and provide a fair comparison between models, the metrics are modified to only use saliency values at human fixation points given that human's do not typically look near the edges of images. By using only human fixation points, any center bias will affect the metrics of each model equally. Additionally, for all models' saliency map output, each frame is normalized between 0 and 1. These metric measurements are similar to the metrics used in Ref. [14], except here we use dynamic video rather than static images.

1) Area Under Curve - Receiver Operating Characteristic: For the AUC-ROC evaluation metric, the saliency map is treated as a binary classifier. By varying the threshold and extracting true positives and false positives at each threshold, an ROC curve can be constructed. Similarly to that used in Ref. [14], [33], for extracting true positives, ground truth eye fixation points for the frame being evaluated are used. For extracting false positives, random eye fixation points from other videos in the dataset are used. This process is repeated 100 times for each frame and the average score is computed, and further, the average is computed across all frames to determine the score for a given video. The final AUC-ROC score is the average of that over each video in the dataset. A score of 1 represents perfect prediction, 0.5 represents chance, and less than 0.5 represents anti-correlation. This score is then normalized by the AUC-ROC score describing the ability of human fixations to predict other human fixations [14], [72].

2) Kullback Leibler Divergence: The KLD evaluation metric is effectively the "difference" between two distributions. The first is a histogram of the saliency values sampled at true fixation points of the frame being evaluated. The second is a histogram of saliency values computed at random fixation points in the same frame. This is modified (similarly to Ref. [14], [33], [69]) such that the random fixation points are taken from randomly selected fixation points from other videos in the dataset. A higher KLD value is better. The same normalization method used in computing the the AUC-ROC score is used for normalizing the KLD score for each video and for each model [72].

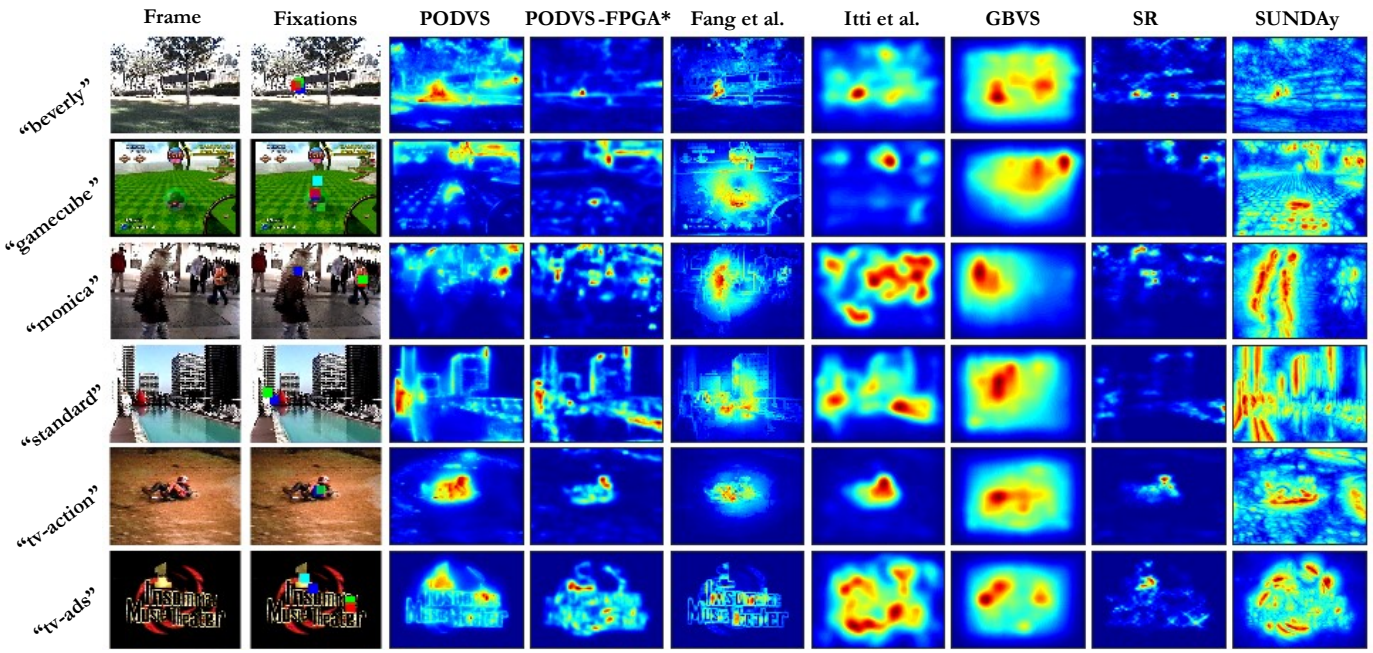


Fig. 4. Comparison of the PODVS and PODVS-FPGA (Hybrid FPGA PODVS implementation) dynamic visual saliency map output with SOTA models. Each row signifies a single frame from a single video from a unique class from the CRCNS dataset noted by the left/vertical labels. The first column, "Frame", is the frame on which saliency was computed. The second column, "Fixations", is the same frame overlaid with colored squares signifying the locations of subjects' fixations at that frame. Different colors correspond to different observers. Columns three through nine are the dynamic saliency map outputs at the frame for our PODVS model (software implementation), our PODVS hybrid FPGA implementation (PODVS-FPGA), the Fang et al. [34], Itti et al. [31], GBVS [36], SR [30], and SUNDAy [33] models, respectively.

VI. Results and Discussion

A. PODVS Model Evaluation

We first compare our model, PODVS, with five SOTA bottom-up, dynamic visual saliency models [30], [31], [33], [34], [36] previously described using the AUC-ROC and KLD metrics. For comparison, we selected models, which were designed to run on video (incorporate motion), are bottom-up, do not require training, and have publicly available code in order to run the model on the dataset and metrics. A visual comparison of the saliency output of the different models can be seen in Fig. 4. While this shows a qualitative comparison between the models, a quantitative comparison is necessary. The comparison of the AUC-ROC and KLD metrics can be seen in Table II. These results are the average AUC-ROC and average KLD scores across all 50 videos in the dataset. The last column shows the p-values after performing a t-test between the PODVS distribution of scores and the corresponding model. A p-value less than 10^{-2} signifies the result is statistically significant. For the AUC-ROC score, all models performed better than chance. It can be seen that our model, PODVS, performed significantly better than all five other SOTA models for the AUC-ROC metric (AUC-ROC $= 0.6745$). It further performed better than all models except for Fang et al. [34] for the KLD metric. This is most likely due to the additional texture feature computed in the Fang et al. model, which is not considered in this model, but can easily be integrated [19]. Studies have shown texture plays an important role in early vision and perception [73]. Further, it has been shown that incorporating texture can improve saliency prediction [19]. The Itti et al. had slightly lower scores (AUC-ROC $= 0.6636$, KLD $= 0.3457$) than our PODVS model, but higher than the other three models. The closeness in performance of our PODVS model and the Itti et al. model may be due to the same normalization operator used in both models. However, our model computes the notion of dynamic proto-objects prior to this normalization operator and performs better, hence, supporting Gestalt psychology and the hypothesis that attention is object-based, not feature-based.

TABLE II
Average AUC-ROC and KLD Scores on CRCNS Dataset

Model	AUC-ROC	p-value	KLD	p-value
Chance	0.5	–	0	–
PODVS	$\mathbf{0.6745 \pm 0.07}$	–	0.3507 ± 0.08	–
PODVS-FIXED[1]	0.6676 ± 0.07	$< 10^{-12}$	0.3501 ± 0.08	$< 10^{-12}$
PODVS-FPGA[2]	0.6511 ± 0.08	$< 10^{-12}$	0.3082 ± 0.08	$< 10^{-12}$
Fang et al. [34]	0.6642 ± 0.07	$< 10^{-12}$	$\mathbf{0.3594 \pm 0.08}$	$< 10^{-12}$
Itti et al. [31]	0.6636 ± 0.10	$< 10^{-12}$	0.3457 ± 0.10	$< 10^{-8}$
GBVS [37]	0.5859 ± 0.11	$< 10^{-12}$	0.3092 ± 0.07	$< 10^{-12}$
SR [30]	0.5595 ± 0.07	$< 10^{-12}$	0.2522 ± 0.07	$< 10^{-12}$
SUNDAy [33]	0.6080 ± 0.09	$< 10^{-12}$	0.3025 ± 0.10	$< 10^{-12}$

[1]PODVS-FPGA software emulation using fixed-point precision (640 × 480 input resolution).
[2]Current FPGA implementation (112 × 84 input resolution).

B. PODVS-FPGA Evaluation

1) Accuracy: To verify the accuracy of the hybrid FPGA implementation, at each step of processing we compared the FPGA implementation output to that of the original software based model. The software based model parameters were set to match those of those used by the FPGA implementation.

Ideally, the output at each stage should be identical, however, the software implementation uses floating-point precision while the FPGA uses fixed-point precision for its computation due to the digital nature of the FPGA hardware. The output at each stage matched up to the error due to precision. A visual comparison between the implementations' saliency map output can be seen in Fig. 4. To evaluate the PODVS-FPGA implementation's ability to predict human eye fixations, we ran the PODVS-FPGA model on the same CRCNS dataset scaled down to 112×84 resolution and compute the same metrics, AUC-ROC and KLD. The PODVS-FPGA model performed similarly to the PODVS software implementation with a AUC-ROC score of 0.6511 and KLD score of 0.3082, only slightly lower due to reduced video resolution and use of fixed-point precision. To evaluate the PODVS-FPGA implementation at full resolution (640×480), we have implemented a software emulation of the PODVS-FPGA model using fixed-point precision. This fixed-point PODVS implementation, PODVS-FIXED, directly emulates the FPGA implementation, which allows for running the model at any resolution. Table II shows the FPGA-FIXED metrics on the CRCNS dataset at 640×480 resolution. With an AUC-ROC score of 0.6676, and a KLD score of 0.3501, we validate that the FPGA implementation's ability to predict eye fixations is comparable to that of the PODVS floating-point software implementation. The small reduction in scores is due to its use of fixed-point precision. Therefore, given an FPGA with sufficient resources, the PODVS-FPGA implementation at full resolution would similarly outperform other SOTA models as does the software floating-point implementation.

2) Resources: The resource utilization by the FPGA for an input resolution of 112×84 can be seen in Table III. The resources utilized are for processing a single channel. The limiting resources were the available DSP (Digital Signal Processing) slices and BRAM. While external DDR3 memory is available, memory access latency is longer than that of internal BRAM, hence we limited memory utilization to BRAM only. Utilizing an FPGA with sufficient BRAM and DSP blocks, all 9 channels can be implemented and processed in parallel. Additionally, Fig. 5 is a plot showing how the total BRAM resources required scales with the input resolution including the BRAM allocated to the various processing steps.

TABLE III
RESOURCES USED BY OK XEM7350-160T FPGA FOR PODVS

Resource	Available	Utilization	Utilization %
Slice Registers	202,800	33,911	16%
Slice LUTs	101,400	36,313	35%
BRAM (B18E1)	650	569	87%
DSP48E1 Slices	600	594	99%

3) Speed: At a resolution of 112×84, processing each of the 9 channels sequentially, the FPGA-based PODVS model has a framerate of $2.08Hz$ for this Opal Kelly FPGA running on a 100MHz clock. The software version of this model (also with equally rescaled parameters) has a runtime of $\sim 1.27s$ on an Intel Quad-Core i7 PC. This is a $2.64\times$ speedup in computation time. This is a significant speedup considering it only processes a single channel in parallel on the FPGA. Given an FPGA with sufficient resources for computing all 9 channels in parallel, this FPGA implementation has a $23.77\times$ speedup (at a framerate of 18.71Hz) with respect to our software implementation. To calculate the amount of sufficient resources, it is assumed that the resources scales roughly linearly with the number of channels being processed in parallel. Therefore, for the 112×84 resolution version, the resource utilization noted in Table III can be multiplied by 9 for estimating the total amount of resources required for parallel processing of all 9 channels. A high-end FPGA with more resources would be sufficient for processing all 9 channels in parallel at either resolution. This significant speedup in dynamic saliency map computation and small-size of the FPGA, makes this FPGA implementation suitable for real-world applications requiring real-time processing. Table IV summarizes these results. Fig. 5 shows how the framerate varies with respect to input resolution when processing a single channel in parallel as well as 9 parallel channels. This framerate includes processing steps occurring on the host (our selected CPU). Having sufficient resources to process all 9 channels in parallel will result in a $\times 9$ faster framerate. It is assumed a 100 MHz FPGA internal clock is used.

TABLE IV
HYBRID FPGA VS. SOFTWARE FRAMERATE @ 112×84

Device	Framerate	Speedup
Kintex-7 XC7K160T-1FFG676C	2.08 Hz	$2.64\times$
Other FPGA*	18.71 Hz	$23.77\times$

* = FPGA with sufficient resources to process 9 channels in parallel.

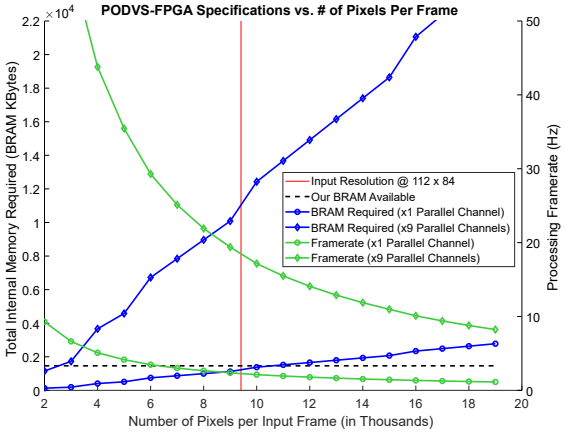


Fig. 5. The left y-axis corresponds with the blue lines, showing how BRAM utilization for a single parallel channel and $\times 9$ parallel channels implemented vary with number of input pixels, i.e., input resolution. The black dotted line signifies the available resources on the Kintex-7 FPGA we used. The right y-axis corresponds with the green lines, showing framerate for a single parallel channel and $\times 9$ parallel channels.

C. Towards an End-to-End FPGA Implementation

The objective of this hybrid FPGA implementation is to demonstrate a speedup in framerate of the PODVS for real-time applications. This requires a speedup in the computationally demanding steps of the model, those being the many

MAC operations. At an input resolution of 112×84, a total of 18.9×10^6 MAC operations are required. Given a 640×480 resolution with full 13×13 kernel sizes, 7.16×10^9 MAC operations are required. It is clear that these MAC operations are the bottleneck of the model, hence, this hybrid FPGA implementation implements all MAC implementations on the FPGA parallel hardware. Ideally, all additional processing should be implemented on hardware as well to have a complete end-to-end FPGA solution. We select an FPGA over GPU because high-end FPGAs have been shown to outperform high-end GPUs in operations/second ($5.4\times$) as well as operations/Watt ($2.3\times$) [74]. Other work has also shown FPGAs to outperform GPUs for applications requiring large number of MAC operations [75], [76]. A full end-to-end FPGA implementation will allow running the PODVS model in real-time under low-SWaP specifications. This is advantageous for applications which can benefit from real-time and low SWaP saliency computation including image quality assessment [77], unmanned aerial vehicle navigation and obstacle avoidance [78], environment exploration for robotics [79], and robot-assisted surgery for learning where to look [80]. This work brings us closer to such an end-to-end implementation.

VII. Conclusion

In conclusion, we report on two advances. First,we present a novel dynamic visual saliency model, PODVS, based on the notion of dynamic proto-objects that exist preattentively within the scene. This neuromorphic model is feed-forward, bottom-up, and biologically plausible in its computation, suggesting how dynamic visual saliency is computed in the early stages of human visual processing. Our neuromorphic model outperforms other SOTA dynamic visual saliency models in predicting human fixations on videos and no training on large datasets is required. Secondly, we present a novel hybrid FPGA implementation for real-time processing of this PODVS model to be used for real-world applications. The hybrid FPGA implementation allows for 18.71 Hz processing, a $\sim 23\times$ speedup compared to that of our current software implementation, while maintaining high similarity in saliency map output. Future work involves implementing the less computationally demanding tasks on hardware as well in order to achieve a full end-to-end FPGA implementation. This work may serve as the foundation for future work, which incorporates dynamic proto-objects computation in the presence of dynamic visual stimuli for higher-level, top-down tasks such as image detection, tracking, and classification in which learning is involved. Furthermore, the biofidelic nature of this work makes this model suitable for processing on neuromorphic hardware in a spike-based manner to further achieve the low SWaP specifications we seek [81], [82].

Acknowledgment

This work was funded and supported by the Office of Naval Research under MURI (Multidisciplinary University Research Initiative) Grant N000141010278, the National Institutes of Health Grant R01EY027544 (CRCNS), and the SERB (Science and Engineering Research Board) Grant, India: ECR/2017/002517.

References

[1] K. Koch, J. McLean, M. Berry, P. Sterling, V. Balasubramanian, and M. A. Freed, "Efficiency of information transmission by retinal ganglion cells," *Current Biology*, vol. 14, no. 17, pp. 1523–1530, 2004.

[2] S. P. Strong, R. Koberle, R. R. d. R. van Steveninck, and W. Bialek, "Entropy and information in neural spike trains," *Physical review letters*, vol. 80, no. 1, p. 197, 1998.

[3] J. Duncan, "Selective attention and the organization of visual information." *Journal of Experimental Psychology: General*, vol. 113, no. 4, p. 501, 1984.

[4] J. P. Gottlieb, M. Kusunoki, and M. E. Goldberg, "The representation of visual salience in monkey parietal cortex," *Nature*, vol. 391, no. 6666, p. 481, 1998.

[5] D. L. Robinson and S. E. Petersen, "The pulvinar and visual salience," *Trends in neurosciences*, vol. 15, no. 4, pp. 127–132, 1992.

[6] F. T. Qiu, T. Sugihara, and R. Von Der Heydt, "Figure-ground mechanisms provide structure for selective attention," *Nature neuroscience*, vol. 10, no. 11, p. 1492, 2007.

[7] S. Park, A. Al Maashri, Y. Xiao, K. M. Irick, and V. Narayanan, "Saliency-driven dynamic configuration of hmax for energy-efficient multi-object recognition," in *VLSI (ISVLSI), 2013 IEEE Computer Society Annual Symposium on*. IEEE, 2013, pp. 139–144.

[8] D. Wang, G. Li, W. Jia, and X. Luo, "Saliency-driven scaling optimization for image retargeting," *The Visual Computer*, vol. 27, no. 9, pp. 853–860, 2011.

[9] M. Riesenhuber and T. Poggio, "Hierarchical models of object recognition in cortex," *Nature neuroscience*, vol. 2, no. 11, p. 1019, 1999.

[10] C. Koch and S. Ullman, "Shifts in selective visual attention: towards the underlying neural circuitry," *Human Neurobiol.*, vol. 4, pp. 219–227, 1985.

[11] 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*, vol. 20, no. 11, pp. 1254–1259, 1998.

[12] Y. Sun, R. Fisher, F. Wang, and H. M. Gomes, "A computer vision model for visual-object-based attention and eye movements," *Computer vision and image understanding*, vol. 112, no. 2, pp. 126–142, 2008.

[13] D. Walther and C. Koch, "Modeling attention to salient proto-objects," *Neural networks*, vol. 19, no. 9, pp. 1395–1407, 2006.

[14] A. F. Russell, S. Mihalaş, R. von der Heydt, E. Niebur, and R. Etienne-Cummings, "A model of proto-object based saliency," *Vision research*, vol. 94, pp. 1–15, 2014.

[15] V. Yanulevskaya, J. Uijlings, J.-M. Geusebroek, N. Sebe, and A. Smeulders, "A proto-object-based computational model for visual saliency," *Journal of vision*, vol. 13, no. 13, pp. 27–27, 2013.

[16] S. Mihalas, Y. Dong, R. von der Heydt, and E. Niebur, "Mechanisms of perceptual organization provide auto-zoom and auto-localization for attention to objects," *Proceedings of the National Academy of Sciences*, vol. 108, no. 18, pp. 7583–8, 2011, pMC3088583.

[17] B. Hu, R. Kane-Jackson, and E. Niebur, "A proto-object based saliency model in three-dimensional space," *Vision Research*, vol. 119, pp. 42–49, 2016, pMID: 26739278.

[18] N. Wagatsuma, R. von der Heydt, and E. Niebur, "Spike Synchrony Generated by Modulatory Common Input through NMDA-type Synapses," *Journal of Neurophysiology*, vol. 116, no. 3, pp. 1418–1433, 2016.

[19] T. Uejima, E. Niebur, and R. Etienne-Cummings, "Proto-object based saliency model with texture detection channel," *Frontiers in computational neuroscience*, vol. 14, p. 84, 2020.

[20] A. Chalbi, J. Ritchie, D. Park, J. Choi, N. Roussel, N. Elmqvist, and F. Chevalier, "Common fate for animated transitions in visualization," *IEEE transactions on visualization and computer graphics*, vol. 26, no. 1, pp. 386–396, 2019.

[21] R. A. Rensink, "The dynamic representation of scenes," *Visual cognition*, vol. 7, no. 1-3, pp. 17–42, 2000.

[22] A. M. Treisman and G. Gelade, "A feature-integration theory of attention," *Cognitive psychology*, vol. 12, no. 1, pp. 97–136, 1980.

[23] J. L. Molin, A. F. Russell, S. Mihalas, E. Niebur, and R. Etienne-Cummings, "Proto-object based visual saliency model with a motion-sensitive channel," in *2013 IEEE Biomedical Circuits and Systems Conference (BioCAS)*. IEEE, 2013, pp. 25–28.

[24] J. L. Molin, R. Etienne-Cummings, and E. Niebur, "How is motion integrated into a proto-object based visual saliency model?" in *2015 49th Annual Conference on Information Sciences and Systems (CISS)*. IEEE, 2015, pp. 1–6.

[25] D. Mahapatra, S. Winkler, and S.-C. Yen, "Motion saliency outweighs other low-level features while watching videos," International Society for Optics and Photonics, pp. 68 060P–68 060P, 2008.

[26] N. C. Thompson, K. Greenewald, K. Lee, and G. F. Manso, "The computational limits of deep learning," *arXiv preprint arXiv:2007.05558*, 2020.

[27] H. Xu, Y. Ma, H.-C. Liu, D. Deb, H. Liu, J.-L. Tang, and A. K. Jain, "Adversarial attacks and defenses in images, graphs and text: A review," *International Journal of Automation and Computing*, vol. 17, no. 2, pp. 151–178, 2020.

[28] R. Rosenholtz, "A simple saliency model predicts a number of motion popout phenomena," *Vision research*, vol. 39, no. 19, pp. 3157–3163, 1999.

[29] D. Gao, V. Mahadevan, and N. Vasconcelos, "On the plausibility of the discriminant center-surround hypothesis for visual saliency," *Journal of vision*, vol. 8, no. 7, pp. 13–13, 2008.

[30] H. J. Seo and P. Milanfar, "Static and space-time visual saliency detection by self-resemblance," *Journal of vision*, vol. 9, no. 12, pp. 15–15, 2009.

[31] L. Itti, "Real-time high-performance attention focusing in outdoors color video streams," in *Human Vision and Electronic Imaging VII*, vol. 4662. International Society for Optics and Photonics, 2002, pp. 235–243.

[32] L. Itti, "Quantifying the contribution of low-level saliency to human eye movements in dynamic scenes," *Visual Cognition*, vol. 12, no. 6, pp. 1093–1123, 2005.

[33] L. Zhang, M. H. Tong, and G. W. Cottrell, "SUNDAy: Saliency using natural statistics for dynamic analysis of scenes," in *Proceedings of the 31st annual cognitive science conference*. AAAI Press Cambridge, MA, 2009, pp. 2944–2949.

[34] Y. Fang, Z. Wang, W. Lin, and Z. Fang, "Video saliency incorporating spatiotemporal cues and uncertainty weighting," *IEEE transactions on image processing*, vol. 23, no. 9, pp. 3910–3921, 2014.

[35] E. Niebur and C. Koch, "Control of Selective Visual Attention: Modeling the "Where" Pathway," in *Neural Information Processing Systems*, D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, Eds. Cambridge, MA: MIT Press, 1996, vol. 8, pp. 802–808.

[36] J. Harel, C. Koch, and P. Perona, "Graph-based visual saliency," in *Advances in neural information processing systems*, 2007, pp. 545–552.

[37] J. Harel, C. Koch, and P. Perona, "A saliency implementation in matlab," *URL: http://www. klab. caltech. edu/harel/share/gbvs. php*, 2006.

[38] S. Marat, T. H. Phuoc, L. Granjon, N. Guyader, D. Pellerin, and A. Guérin-Dugué, "Modelling spatio-temporal saliency to predict gaze direction for short videos," *International journal of computer vision*, vol. 82, no. 3, pp. 231–243, 2009.

[39] C. Guo and L. Zhang, "A novel multiresolution spatiotemporal saliency detection model and its applications in image and video compression," *IEEE Transactions on Image Processing*, vol. 19, no. 1, pp. 185–198, 2010.

[40] V. Leboran, A. Garcia-Diaz, X. R. Fdez-Vidal, and X. M. Pardo, "Dynamic whitening saliency," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 5, pp. 893–907, 2016.

[41] S. M. Muddamsetty, D. Sidibé, A. Trémeau, and F. Mériaudeau, "Salient objects detection in dynamic scenes using color and texture features," *Multimedia Tools and Applications*, vol. 77, no. 5, pp. 5461–5474, 2018.

[42] F. Zhou, S. Bing Kang, and M. F. Cohen, "Time-mapping using space-time saliency," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 3358–3365.

[43] C. Bak, A. Kocak, E. Erdem, and A. Erdem, "Spatio-temporal saliency networks for dynamic saliency prediction," *IEEE Transactions on Multimedia*, vol. 20, no. 7, pp. 1688–1698, 2017.

[44] T. Liu, Z. Yuan, J. Sun, J. Wang, N. Zheng, X. Tang, and H.-Y. Shum, "Learning to detect a salient object," *IEEE Transactions on Pattern analysis and machine intelligence*, vol. 33, no. 2, pp. 353–367, 2011.

[45] T. Chen, L. Lin, L. Liu, X. Luo, and X. Li, "Disc: Deep image saliency computing via progressive representation learning," *IEEE transactions on neural networks and learning systems*, vol. 27, no. 6, pp. 1135–1149, 2016.

[46] W. Wang, J. Shen, F. Guo, M.-M. Cheng, and A. Borji, "Revisiting video saliency: A large-scale benchmark and a new model," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4894–4903.

[47] N. Liu, J. Han, T. Liu, and X. Li, "Learning to predict eye fixations via multiresolution convolutional neural networks," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 2, pp. 392–404, 2016.

[48] C.-S. Bouganis, P. Y. Cheung, and L. Zhaoping, "Fpga-accelerated pre-attentive segmentation in primary visual cortex," in *2006 International Conference on Field Programmable Logic and Applications*. IEEE, 2006, pp. 1–6.

[49] Z. Li, "Visual segmentation by contextual influences via intra-cortical interactions in the primary visual cortex," *Network: computation in neural systems*, vol. 10, no. 2, pp. 187–212, 1999.

[50] S. Kestur, M. S. Park, J. Sabarad, D. Dantara, V. Narayanan, Y. Chen, and D. Khosla, "Emulating mammalian vision on reconfigurable hardware," in *Field-Programmable Custom Computing Machines (FCCM), 2012 IEEE 20th Annual International Symposium on*. IEEE, 2012, pp. 141–148.

[51] P. Akselrod, F. Zhao, I. Derekli, C. Farabet, B. Martini, Y. LeCun, and E. Culurciello, "Hardware accelerated visual attention algorithm," in *Information Sciences and Systems (CISS), 2011 45th Annual Conference on*. IEEE, 2011, pp. 1–6.

[52] B. Kim, H. Okuno, T. Yagi, and M. Lee, "Implementation of visual attention system using artificial retina chip and bottom-up saliency map model," in *International Conference on Neural Information Processing*. Springer, 2011, pp. 416–423.

[53] A. Moradhasel, B. N. Araabi, S. M. Fakhraie, and M. N. Ahmadabadi, "Fast saliency map extraction from video: A hardware approach," in *Machine Vision and Image Processing (MVIP), 2013 8th Iranian Conference on*. IEEE, 2013, pp. 64–69.

[54] F. Barranco, J. Diaz, B. Pino, and E. Ros, "Real-time visual saliency architecture for fpga with top-down attention modulation," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 3, pp. 1726–1735, 2014.

[55] S. Bae, Y. C. P. Cho, S. Park, K. M. Irick, Y. Jin, and V. Narayanan, "An fpga implementation of information theoretic visual-saliency system and its optimization," in *Field-Programmable Custom Computing Machines (FCCM), 2011 IEEE 19th Annual International Symposium on*. IEEE, 2011, pp. 41–48.

[56] N. Bruce and J. Tsotsos, "Attention based on information maximization," *Journal of Vision*, vol. 7, no. 9, pp. 950–950, 2007.

[57] "CSMS lab - JHU," https://github.com/csmslab/dynamic-proto-object-saliency.

[58] H. Zhou, H. S. Friedman, and R. Von Der Heydt, "Coding of border ownership in monkey visual cortex," *Journal of Neuroscience*, vol. 20, no. 17, pp. 6594–6611, 2000.

[59] R. L. De Valois, N. P. Cottaris, L. E. Mahon, S. D. Elfar, and J. A. Wilson, "Spatial and temporal receptive fields of geniculate and cortical cells and directional selectivity," *Vision research*, vol. 40, no. 27, pp. 3685–3702, 2000.

[60] P. Cavanagh, "Attention-based motion perception," *Science*, vol. 257, no. 5076, pp. 1563–1565, 1992.

[61] M. Hawken, R. Shapley, and D. Grosof, "Temporal-frequency selectivity in monkey visual cortex," *Vis Neurosci*, vol. 13, no. 3, pp. 477–492, 1996.

[62] D. Parkhurst, "Selective attention in natural vision: using computational models to quantify stimulus-driven attentional allocation," Ph.D. dissertation, The Johns Hopkins University, 2002.

[63] E. Craft, H. Schütze, E. Niebur, and R. Von Der Heydt, "A neural model of figure–ground organization," *Journal of neurophysiology*, vol. 97, no. 6, pp. 4310–4326, 2007.

[64] L. Itti and R. Carmi, "Eye-tracking data from human volunteers watching complex video stimuli," 2009. [Online]. Available: http://dx.doi.org/10.6080/K0TD9V7F

[65] Z. Bylinskii, T. Judd, A. Oliva, A. Torralba, and F. Durand, "What do different evaluation metrics tell us about saliency models?" *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 3, pp. 740–757, 2018.

[66] D. M. Green, J. A. Swets *et al.*, *Signal detection theory and psychophysics*. Wiley New York, 1966, vol. 1.

[67] L. Itti and P. Baldi, "A principled approach to detecting surprising events in video," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1. IEEE, 2005, pp. 631–637.

[68] L. Itti and P. F. Baldi, "Bayesian surprise attracts human attention," in *Advances in neural information processing systems*, 2006, pp. 547–554.

[69] B. W. Tatler, "The central fixation bias in scene viewing: Selecting an optimal viewing position independently of motor biases and image feature distributions," *Journal of vision*, vol. 7, no. 14, pp. 4–4, 2007.

[70] F. Vitu, Z. Kapoula, D. Lancelin, and F. Lavigne, "Eye movements in reading isolated words: Evidence for strong biases towards the center of the screen," *Vision research*, vol. 44, no. 3, pp. 321–338, 2004.

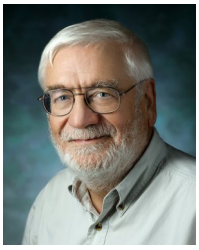
[71] D. Parkhurst and E. Niebur, "Scene content selected by active vision," *Spatial vision*, vol. 16, no. 2, pp. 125–154, 2003.

[72] C. M. Masciocchi, S. Mihalas, D. Parkhurst, and E. Niebur, "Everyone knows what is interesting: Salient locations which should be fixated," *Journal of vision*, vol. 9, no. 11, pp. 25–25, 2009.

[73] J. R. Bergen and E. H. Adelson, "Early vision and texture perception," *Nature*, vol. 333, no. 6171, pp. 363–364, 1988.
[74] E. Nurvitadhi, G. Venkatesh, J. Sim, D. Marr, R. Huang, J. Ong Gee Hock, Y. T. Liew, K. Srivatsan, D. Moss, S. Subhaschandra *et al.*, "Can fpgas beat gpus in accelerating next-generation deep neural networks?" in *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, 2017, pp. 5–14.
[75] S. I. Venieris and C.-S. Bouganis, "fpgaconvnet: Mapping regular and irregular convolutional neural networks on fpgas," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 2, pp. 326–342, 2018.
[76] G. Orchard, J. G. Martin, R. J. Vogelstein, and R. Etienne-Cummings, "Fast neuromimetic object recognition using fpga outperforms gpu implementations," *IEEE transactions on neural networks and learning systems*, vol. 24, no. 8, pp. 1239–1252, 2013.
[77] W. Zhang, A. Borji, Z. Wang, P. Le Callet, and H. Liu, "The application of visual saliency models in objective image quality assessment: A statistical evaluation," *IEEE transactions on neural networks and learning systems*, vol. 27, no. 6, pp. 1266–1278, 2015.
[78] Z. Ma, C. Wang, Y. Niu, X. Wang, and L. Shen, "A saliency-based reinforcement learning approach for a uav to avoid flying obstacles," *Robotics and Autonomous Systems*, vol. 100, pp. 108–118, 2018.
[79] C. Craye, D. Filliat, and J.-F. Goudou, "Environment exploration for object-based visual saliency learning," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 2303–2309.
[80] M. Islam, Y. Li, and H. Ren, "Learning where to look while tracking instruments in robot-assisted surgery," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2019, pp. 412–420.
[81] C. S. T. Thakur, J. Molin, G. Cauwenberghs, G. Indiveri, K. Kumar, N. Qiao, J. Schemmel, R. M. Wang, E. Chicca, J. Olson Hasler *et al.*, "Large-scale neuromorphic spiking array processors: A quest to mimic the brain," *Frontiers in neuroscience*, vol. 12, p. 891, 2018.
[82] C. S. Thakur, J. L. Molin, T. Xiong, J. Zhang, E. Niebur, and R. Etienne-Cummings, "Neuromorphic visual saliency implementation using stochastic computation," in *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2017, pp. 1–4.

Jamal Lottier Molin Dr. Jamal Lottier Molin received the B.S. degree in computer engineering from University of Maryland, Baltimore County (UMBC), Baltimore, MD, USA, in 2011, and the M.S.E and Ph.D. degrees in electrical and computer engineering from the Johns Hopkins University, Baltimore, MD, USA, in 2015 and 2017. At UMBC, he was a recipient of the Meyerhoff Scholarship. He also received the DoD SMART fellowship. He currently works for Northrop Grumman doing research in the field of neuromorphic engineering for vision applications.

Chetan Singh Thakur Dr. Chetan Singh Thakur is an Assistant Professor at Indian Institute of Science (IISc), Bangalore, and has an Adjunct Faculty appointment at the International Center for Neuromorphic Systems, Western Sydney University (WSU), Australia. Dr. Chetan Singh Thakur (Senior Member, IEEE) received his Ph.D. in neuromorphic engineering at the MARCS Research Institute, Western Sydney University in 2016. He then worked as a research fellow at the Johns Hopkins University. In addition, Dr. Thakur has extensive industrial experience. He worked for 6 years with Texas Instruments Singapore as a senior Integrated Circuit Design Engineer, designing IPs for mobile processors. His research expertise lies in neuromorphic computing, mixed-signal VLSI systems, computational neuroscience, probabilistic signal processing, and machine learning. His research interest is to understand the signal processing aspects of the brain and apply those to build novel intelligent systems. He is recipients of several awards such as Young Investigator Award from Pratiksha Trust, Early Career Research Award by Science and Engineering Research Board- India, Inspire Faculty Award by Department of Science and Technology- India.



Ernst Niebur Dr. Ernst Niebur received the M.S. degree (Diplom Physiker) from the Universität Dortmund, Dortmund, Germany, the Postgraduate Diploma in artificial intelligence from the Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland, and the Ph.D. degree (Dr ès sciences) in physics from the Université de Lausanne, Lausanne, Switzerland. His dissertation topic was a detailed computational model of the motor nervous system of the nematode C. elegans. He was a Research Fellow and a Senior Research Fellow at the California Institute of Technology, Pasadena, CA, USA, and an Adjunct Professor of Information Technology and of Life Sciences at the Queensland University of Technology, Brisbane, Australia. He joined the faculty of Johns Hopkins University, Baltimore, MD, USA, in 1995, where he is currently a Professor of Neuroscience in the School of Medicine, and of Brain and Psychological Sciences in the School of Arts and Sciences. He is also Affiliated Faculty of the Institute for Computational Medicine, Whiting School of Engineering. He uses computational neuroscience to understand the function of the nervous system at many levels. Prof. Niebur is the recipient of a Seymour Cray (Switzerland) Award in Scientific Computation, an Alfred P. Sloan Fellowship, and a National Science Foundation CAREER Award.

Ralph Etienne-Cummings Dr. Ralph Etienne-Cummings (F'13) received the B.S. degree in physics from Lincoln University, Lincoln, PA, USA, in 1988, and the M.S.E.E. and Ph.D. degrees in electrical engineering from the University of Pennsylvania, Philadelphia, PA, USA, in 1991 and 1994, respectively. He is currently a Professor of electrical and computer engineering, and computer science with The Johns Hopkins University, Baltimore, MD, USA. He was the Founding Director of the Institute of Neuromorphic Engineering. He has authored more than 200 peer-reviewed articles and holds numerous patents. He has served as the Chairman of various IEEE Circuits and Systems (CAS) Technical Committees and was elected as a member of CAS Board of Governors. He also serves on numerous editorial boards. He is a recipient of the NSFs Career and Office of Naval Research Young Investigator Program Awards. He was a Visiting African Fellow at the University of Cape Town, Fulbright Fellowship Grantee, Eminent Visiting Scholar at the University of Western Sydney and has also received numerous publication awards, including the 2012 Most Outstanding Paper of the IEEE Transactions on Neural Systems and Rehabilitation Engineering. In addition, he was recently recognized as a Science Maker, an African American history archive.