Exploring gaze behavior through Object Detection and Saliency Maps for walking pedestrians with *Homonymous Hemianopia*

DHRUV SHETH[1*], AYUSH KUMAR[2*], SHRINIVAS PUNDLIK[2], AND GANG LUO[2]

[1] *California Institute of Technology*
[2] *Schepens Eye Research Institute, Harvard Medical School*
[*] *These Authors contributed equally to this work.*

Compiled January 22, 2023

Understanding gaze behavior in individuals with *Homonymous Hemianopia* (HH)—a visual field loss caused by occipital lobe damage—requires nuanced analysis of visual attention in real-world scenarios. This study quantifies gaze deviations in HH patients during outdoor walks, employing a portable eye-tracking system integrated with inertial measurement units (IMUs). Analyzing over 56,000 frames of visual data, we utilized object detection frameworks (YOLOv4 Darknet and Faster R-CNN) to classify and track gaze-relevant objects in dynamic settings. A saliency attentive model, enhanced with persistent homology, revealed deviations in fixation priorities, highlighting altered spatial attention hierarchies in HH. Key insights include a bias toward peripheral object attention and reduced focus on centrally salient regions during complex tasks like street crossing. By combining computational tools with behavioral analysis, this framework offers a reliable, non-invasive alternative for diagnosing gaze behaviors in visual impairments and lays the groundwork for adaptive, patient-specific interventions. © 2024 Optica Publishing Group

http://dx.doi.org/10.1364/XX.XX.XXXXXX

1. INTRODUCTION

Examining information in temporal gaze patterns can reveal insights into gaze behaviour essential for studying patterns in visual defect conditions. These patterns provide insight into how a gaze changes it's spatial location and orientation under a landscape and the reason why a specific gaze is focused on a particular set of objects for a larger period of time. The notion here is that it changes widely in patients concerning different visual defects and specifically examining the patterns exhibited in different visual defects may potentially be of clinical relevance for diagnosis and prognosis. Specifically examining patterns in patients with Homonymous Hemianopia allows extrapolating temporal trends to other visual defects as well. Quantitative analysis of these patterns are central to drawing reasonable inferences in behavioral patterns associated with each visual defect. Such an autonomous and non-invasive approach for identification of behavioral patterns provides a low-cost alternative in domains where MRI is not readily accessible or required.

Homonymous Hemianopia is a visual defect associated with the damage of the occipital lobe and the visual cortex leading to loss in visual information from the contralateral visual field. Previous studies focusing on oculography focused on manually observing quantitative patterns using approximated methods to classify them - Meienberg et al. [1]. These studies observed patients to consciously or unconsciously exhibit specific common strategies to spatially locate and fixate objects observed through gaze patterns. This helped draw assumptions that locating and fixating strategies are specific to visual defects and can be used to identify the defect. However, we still observe limitations in oculographic methods to manually classify patterns in longer temporal-trends and hence we propose this Research to employ gaze estimation for localizing quantifiable patterns temporally.

Comparing strategies previously used for diagnosis and prognosis of visual defects is significantly distinctive specifying the evaluation strategies for gaze checks which commonly rely on rapid vibration or change in object around a fixation point throughout the focal view to identify signs of visual defects, in this case Homonymous Hemianopia. Evaluation experiments relying on acquisition of visual stimuli specific to tracking an object around the fixation point does not give insight into behavioural analysis of the same patient under contemporary real world conditions. The following research develops a data acquisition system tracking gazes as well as relative head angles and spatial movements for a Homonymous Hemianopic patient walking in locations namely classified as empty street walking which involves gaze directional towards objects of other classes than 'person' evaluating the nature of objects and trends in gaze over specific object characteristics, crosswalks or lane-crossing mainly to evaluate simultaneous attention distribution over vehicles, people and crosswalk over the focal view, under stationary conditions assess-

ing trends in tendency to observe objects under a certain quadrant of the focal view (characteristic distinctive to Homonymous Hemaniopia) and other similar instances.

The task of calculating gaze extends to the standard gaze estimation method under fixation point experiments where the head position is fixed as a function of time. Under simulated real-world conditions with moving head position in the coordinate space, gaze estimation is achieved through eye gaze estimation relative to the head spatial location and angle, and calculation of head coordinates in space relative to the ground frame. This creates an ensemble of both these calculation algorithms working simultaneously to achieve accurate gaze estimation for objects irrespective of the orientation of head in space. The calculation of head location is based on the information retrieved through IMU sensors positioned on the Homonymous Hemaniopic patient.

Temporally, the gaze is tracked and estimated for the person in-motion using a camera positioned to capture the gaze; simultaneously another camera positioned horizontally outwards captures the focal view of the landscape to correspond to the objects being observed by the gaze. Concerning studies involving mobile subjects with head and gaze position varying with time, most research focused solely on tracking gaze in-head orientation without considering the impact of head orientation over eye-gaze results. (Fotios, Uttley et al.[2], Li, Munn et al.[3]). Since head-orientation wasn't taken into account, an accurate picture of what the eye-gaze is directed towards could not be made. Relative analysis to some-extent may reveal important results; however an absolute estimation of spatial coordinates helps decipher accurate results with respect to the ground frame. Recent improvements are observed in tracking by estimating spatial positions of head movement with respect to confined environments using walking simulators of head-in-space motion capture systems (Barabas, Goldstein et al., [3]; Bowers, Ananyev et al., [4]; Cesqui, de Langenberg et al., [5])

Considering the case of Homonymous Hemianopia can be observed as uni-hemispherical peripheral visual field loss, quantification of temporal patterns in data through gaze tracking provides important behavioural insights which possibly might not be accomplished through eye tracking alone. According to Luo et al. [6], patients with tunnel vision observed saccadic eye-movements large enough to match those of normal vision person under specific conditions while walking outdoors. This makes it hard to distinguish between specific cases in people with visual defects compared to those with normal vision under different environments and using only in-head eye-movements. Extrapolating such data might not yield accurate results and might also contradict findings of previous experiments. Contrary to the frequent methods published to solely identify Homonymous Hemianopia, this paper provides a deeper insight in the behavioural and temporal gaze patterns observed for the patients under various outdoor conditions.

2. DATA ACQUISITION

Our portable gaze tracking system includes a commercial eye tracking system from Positive Science (Positive Science, New York City, NY; 2013) and two commercial IMUs from VectorNav (VectorNav, Dallas, TX). The components are integrated into a lightweight backpack for comfort during wear. The head tracking system is capable of measuring the yaw, pitch, and roll of the head relative to the body trunk, but for this research, we will be focusing on yaw measurement as it is essential for street walking and road crossing. The system is adaptable to measure pitch movements as well. The gaze tracker records two videos, one of the eye and the other of the forward scene, and records the output from the motion sensors. All data streams are logged at a frequency of 30 Hz.

A. Eye Tracking System

The Positive Science eye tracker consists of a camera that tracks one of the eyes, along with a scene camera that captures a 608 field of view, both mounted on one side of a spectacle frame. This design allows for prescription lenses to be fitted in front of both eyes and for optical devices, such as peripheral prism glasses for Homonymous Hemianopia, to be placed on the other eye that is not being tracked. The mobile eye and head tracking system features two data logging computers mounted on a backpack and a front view of the system showcases the eye tracking headgear, which includes the eye and scene cameras, as well as the body and head tracking sensors. The eye tracking system undergoes a 13-point calibration at the beginning and end of each recording session to detect any movement of the headgear during the walk. The calibration is performed at a 4 meter distance and the eye tracker is verified using fixation targets on a whiteboard. If the calibrations match, the same one can be used throughout the sequence, but if they don't, the sequence is split into two parts, with each part using a different calibration. Convenient locations are identified along the walking route for brief calibration checks and the subject is asked to fixate on specific points while audio is recorded to assist with identifying the calibration points during post-processing. If the eye tracker accurately locates the point of regard, the pre-walk calibration is considered valid. If not, the calibration performed at the end is verified and the best calibration is applied to the corresponding segment. If neither calibration is found to be valid, the segments are discarded from analysis.

B. Head Tracking System

We use Inertial Measurement Units (IMUs) for head tracking instead of video-based cameras because they are simpler to process. IMUs consist of accelerometers, gyroscopes, magnetometers, and a microcontroller. Gyroscopes are known for drifting over time, but the drift is corrected with the help of magnetometers and Kalman filters. IMUs can perform well in most environments but can be affected by electromagnetic and metallic objects in busy downtown areas. To address this issue, we employ various strategies including using a pair of matched IMUs to cancel out external electromagnetic interference.

We evaluated various commercial IMUs and selected the VN-100 model from VectorNav based on its accuracy and internal processing capabilities. Although our results were based on this specific IMU, the proposed method can be used with other commercial IMUs. One sensor is attached to a hat visor worn by the subject while the other is attached to the waist, acting as a reference. Both sensors are connected to a laptop in a backpack. Using two sensors, with one on the waist, we can measure horizontal head rotations relative to the body by calculating the difference between the two sensors. It's important to use sensors that are as similar as possible to reduce noise and benefit from the differential signal. External interference affecting both sensors will be canceled out when computing the differential signal, resulting in a cleaner signal of head orientation relative to the body.

Even though the differential signal is used, there may still be drift between the outputs of both sensors due to uncorrelated noise. The drift could be caused by uncontrolled external factors that affect each sensor differently or by internal noise that is unique to each sensor. To minimize the errors caused by uncorrelated noise, our experimental protocol includes several reset operations at preset checkpoints along the outdoor route. These checkpoints are positioned based on the distance walked and are best set after turning points on the walking path. At the checkpoints, the subject is instructed to stop

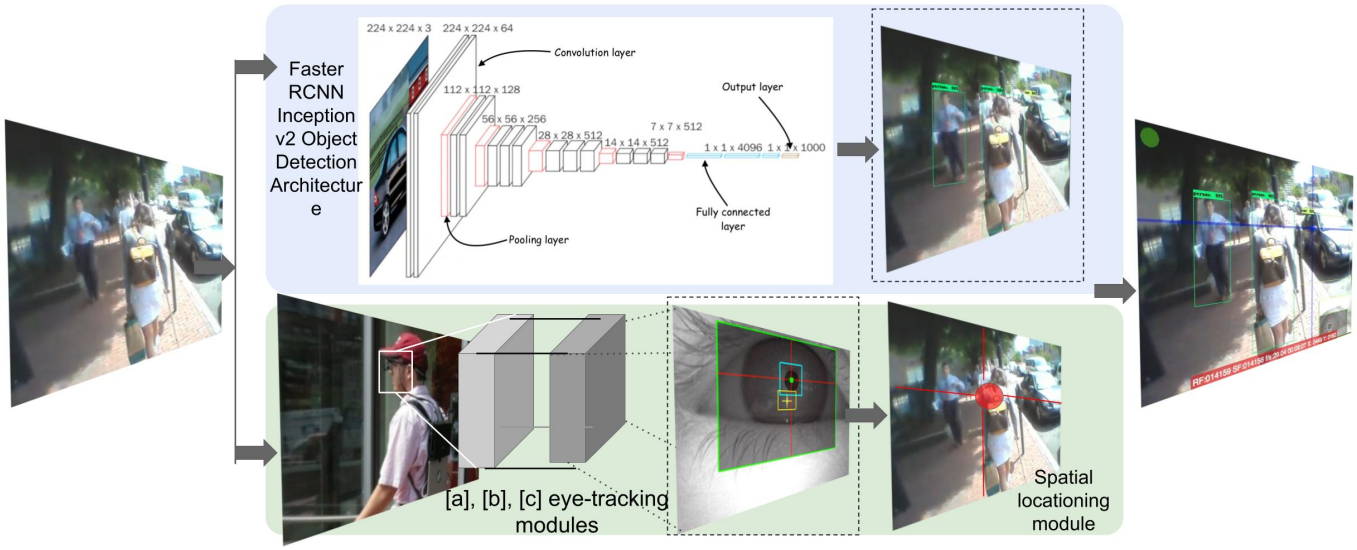

Fig. 1. Designed methodology of the proposed research. Here, modules [a], [b], [c] denotes a feature detection module, thresholding module and realtime pupil and optional corneal reflection tracking module which form an ensemble network for eye-gaze extraction. A Faster RCNN Inception V2 Object Detection framework is used for detecting objects for identifying salient gazed objects.

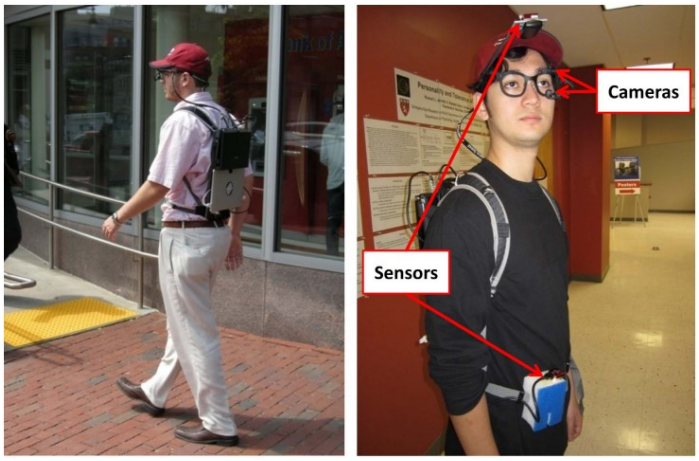

Fig. 2. The system for tracking eye and head movements while mobile is shown in the back and front views. The back view displays the two data logging computers on the backpack that record eye and head movements, and the eye movement logger records video from the scene camera. The front view shows the eye tracker headgear, which contains the eye tracking and scene cameras, and the body and head tracking sensors. Tomasi et al.

briefly and look forward, and a heading reset command is sent to the logging computer to align the orientations of the two sensors and minimize any potential drift. The eye tracker calibration, which was previously described, is also quickly verified at this time. We have created custom software using the IMU software development kit to log the inertial data. This software handles basic sensor diagnostics, parameter setting, and user interaction. The software runs on a lightweight ASUS Eee PC notebook with an Intel Atom N455 processor, which is mounted on the backpack (as shown in Figure 1a). The software outputs a text file that includes the timestamp in seconds, orientation in degrees, acceleration in meters per second squared, and angular velocity in degrees per second for both sensors. The software also keeps track of the reset points and can record a key code from the experimenter to identify significant events.

3. METHOD

One of our main aims in this research is to to analyse and recognize the qualitative nature of the objects guided by a gaze. Here, we make use of Object detection to detect and classify objects based on their extrinsic properties and use these qualitative temporal patterns to identify Homonymous Hemianopic bio-markers. Object detection has been at the epicentre for different computer vision applications. Over the years, Object detection and classification frameworks have been widely used over a spectrum of applications from Autonomous Driving Assistance Systems (ADAS), Healthcare, Robotics etc. and contextually, the mode of deployment of the developed framework varies as well which can be observed in the form of two subsequent algorithms performing similar task extracting completely different patterns of data. Essentially, this research focuses on an Object detection model with a notion of saliency to understand gaze-aware relative importance of different classes and its demographics. The post-processing of our results focuses on identifying relative saliency of detected object classes distinguishing between observant behavioural patterns in Homonymous Hemianopic patients.

We start by comparing different Object detection models suited for our research environment and going with the one which yields

Fig. 3. Comparison of 31/6500 frame from all networks to visually understand comments based on subjective comparison from **??**

Framework	Speed(ms)	COCO mAP	Comments
Faster R-CNN Inception v2	58	28	Efficient to Deploy, Accurate and low latency on the given input video sequence.
SSDlite Mobilenet V2 COCO	27	22	Efficient and real-time however not as accurate results .
SSD Inception V2 COCO	42	24	Results comparable to Faster R-CNN Inception, however few missed objects in occluded scenes
SSD Mobilenet V1 COCO - quantized	30	21	Slightly depreciated performance as compared to SSDlite Mobilenet V2, however lower latency.
Faster R-CNN Resnet50 COCO	89	30	Not efficient enough and multiple false-negatives for other classes, prominent flickering between frames.
Yolo V4 Darknet	73	47.5	Efficient, however produces few false negatives and performs worse in occluded scenarios

Table 1. Initial Objective and Subjective comparison of all tested frameworks on the HMS Homonymous Hemanopia Dataset. As observed in the next section, YOLOv4 Darknet and Faster R-CNN Inception v2 performs better than most other compared.

the most accurate detections specific to the objects in our curated dataset. The detected objects are classified as whether they are gazed or not using three different levels of classification algorithms namely a Euclidean thresholding method through the centroid of the detecting bounding boxes and the other two being geometrical approaches calculating whether a gaze fixation at a spatial frame exists inside the bounding box temporally and the last one being a combination of the previous two; using a Euclidean thresholding method for the foot of perpendicular from the spatial gaze coordinates onto each localised segment of the bounding box using the distance for the perpendicular as the threshold for classification. This will be elaborated upon later.

Initially in this research, we gauge a comparison between 6 different Object detection frameworks each trained on the COCO (Common objects in context) dataset namely YOLO v4 Darknet (You Only Look Once), Faster R-CNN Resnet 50 COCO, SSD Mobilenet V1 COCO- (quantized), SSDlite Mobilenet V2 COCO, SSD Incpetion V2 COCO, Faster R-CNN Inception V2 COCO. It must be noted here that we gauge the frameworks such that a trade-off for speed or processing time for accuracy is preferred in each case. This is because

the research focuses on evaluating behavioural patterns and this does not require a real-time on-device processing out-of-the-box and hence post-processing accurate methods (even though computationally heavy) are preferred. Additionally, even though we try to minimise false negatives as well as false positives in our framework, there is a general preference towards having no-detections in any given frame rather than false-positives since those might yield arbitrary conclusions which might drive our findings onto a different tangent which is avoided. The models trained on COCO dataset are preferred here since the classes in the set of COCO list are representative of most common objects influencing major proportion of behavioural patterns reflective of the visual information perceived. Withing this set of COCO classes, we primarily focus on 24 classes overly dominant in each spatial frame combined temporally which primarily constitute the person class, 4 major *vehicle* classes, road signs and symbols classes and a few other influencing gaze patterns. We use a subjective method of ranking different Object detection frameworks relative to our use-case by observing:

- The accuracy of detected objects in major classes.
- Consistency of the detected objects over temporal frames
- Performance of occluded regions of objects
- Percentage of false-negatives observed in random sample frames and overall calculating the information retrieved through the detected objects overall.

Additionally, we consider the quantitative results such as mAP for the dataset (COCO) and speed(ms) in case two frameworks observe identical performance. Even though in the short term we do not intend on using a real-time Object detection framework for analysing behavioral patterns, we still include SSD Mobilenet v1 and SSDlite Mobilenet *v*2 COCO for understanding their relative performance on this dataset for future studies which are real-time Object detection models which can be deployed on an edge-device.

A. Subjective Evaluation of Frameworks

According to our subjective assertion in **??**, we narrow our focus on comparing the two most optimum frameworks i.e. YOLO v4 Darknet and Faster R-CNN Inception v2 COCO by evaluating the subjective comments in the table. Narrowing our focus down to two frameworks after subjectively and objectively testing the others helps eliminate any other ambiguity in our evaluation. Despite both the frameworks being trained on the COCO dataset, the object detection method both employ widely varies according to the application and in this case, both frameworks tend to vary in their performance depending on the tasks. Assigning each task a weightage according to relative importance and then deciding the most optimum framework based on our discretion 3.

A.1. YOLO V4 Darknet

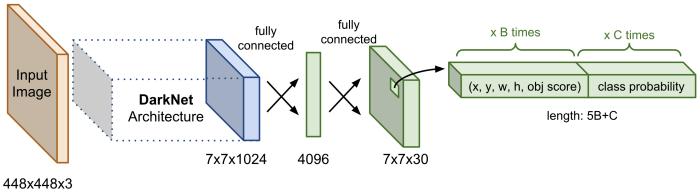

Fig. 4. A generalized network architecture for YOLO

YOLOv4 is a single stage detector framework with it's core of target detection algorithm in it's efficient calculation and small size. The model is capable of effectively localizing the objects in an image which makes it efficient in various applications for object detection. YOLO V4 skips the region proposal network in it's algorithm and tends towards making a trade-off for efficient and faster detection by running over a dense sampling of all possible locations using a one-stage object detection approach. In theory this might impact the performance to some margin, however the magnitude of this difference varies with different applications. YOLO's efforts in minimizing the overlapping of detection boxes lies in using the global image for detection, which encodes the global information reducing the detections of background as an object. In general, YOLO v4 Darknet outperforms most other networks in scenarios where the object is distant in the focal view and has a clear consistent visible region temporally over the frames. Additionally, YOLO is known for its consistency in detecting objects in temporal frames which preserve spatial appearance.

- **Backbone** - CSPDarknet53 is a modified version of the Darknet-53 network. The number of convolution layers in the network are represented by the number '53' and CSP stands for Cross-stage-partial connections.
- **Neck:** Path Aggression Network (PAN) and Spatial Pyramid Pooling (SPP) form the components of the neck of the network. PAN serves as the method of parameter aggregation for different detector levels and SPP is used to increase the receptive field by a significant amount which separates the most significant context features without compromising the network operaton speed.
- **Head:** YOLOv3 is used as the end of the chain object detector for dense predictions and detections.

In general, various different versions of YOLO have shown high computation speed and real-time inference, however since our application focuses on accuracy over real-time detection, the real-time factor isn't taken into consideration while choosing frameworks. The YOLO algorithm is designed to employ an image into a grid of multiple cells and the confidence scores inside the region of the bounding box are predicted for each cell and assigned a class probability. This is done by using IOU metric (Intersection over Union), a common method for evaluating semantic segmentation models, which measure the extent of overlap between detected object and the defined ground truth as a fraction of the total area spanned by the union of the two. Looking at a brief cycle of improvements in the YOLO algorithm, YOLOv2 [7] consisted of anchor boxes- a pre-determined set of boxes to predict the offsets from these pre-defined anchor boxes rather than directly predicting the bounding box. However, YOLOv3 [8] included bounding box prediction over different scales and the inclusion of 53 layers in Darknet. Currently, the most optimum of all proposed, YOLOv4 [9] was superior in terms of both speed and accuracy. As seen in **??**, YOLOv4 provided 47.5% mAP over MS COCO Dataset at a speed of 73ms per frame. In theory, YOLOv4 can be summarised as follows:

It might be further worth noting the workflow of YOLO which is a one-stage detector explained through 6:

- An image is split into a grid of NxN cells, where locally those cells in which the object's centre is located, are responsible for the detection of the object. Associated with each cell is a location of B bounding boxes, confidence score and the probability of an object class dependent on the existence of an object in the bounding box.

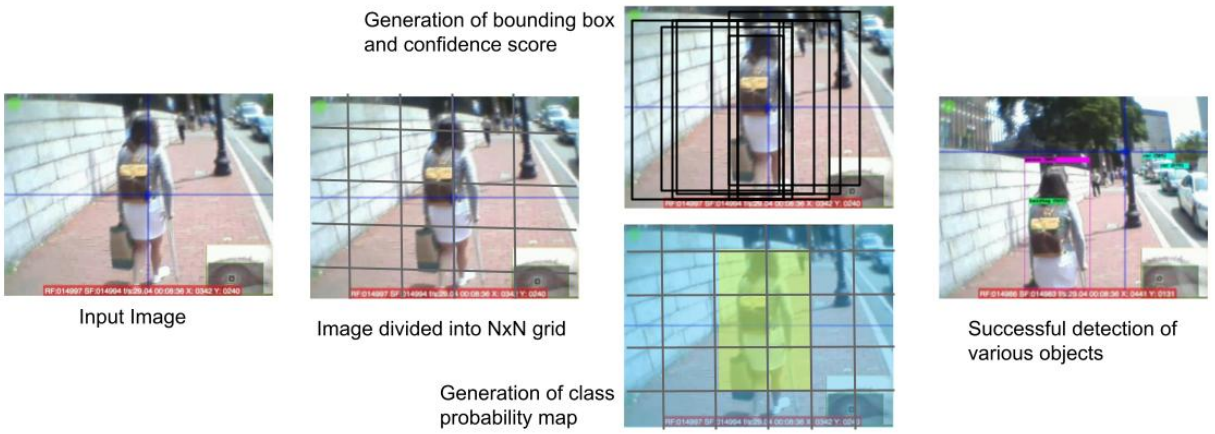

Fig. 5. Different stages of object detection explained through A.1 for a standard one-stage YOLO detection algorithm, further generalized to YOLOv4 Darknet.

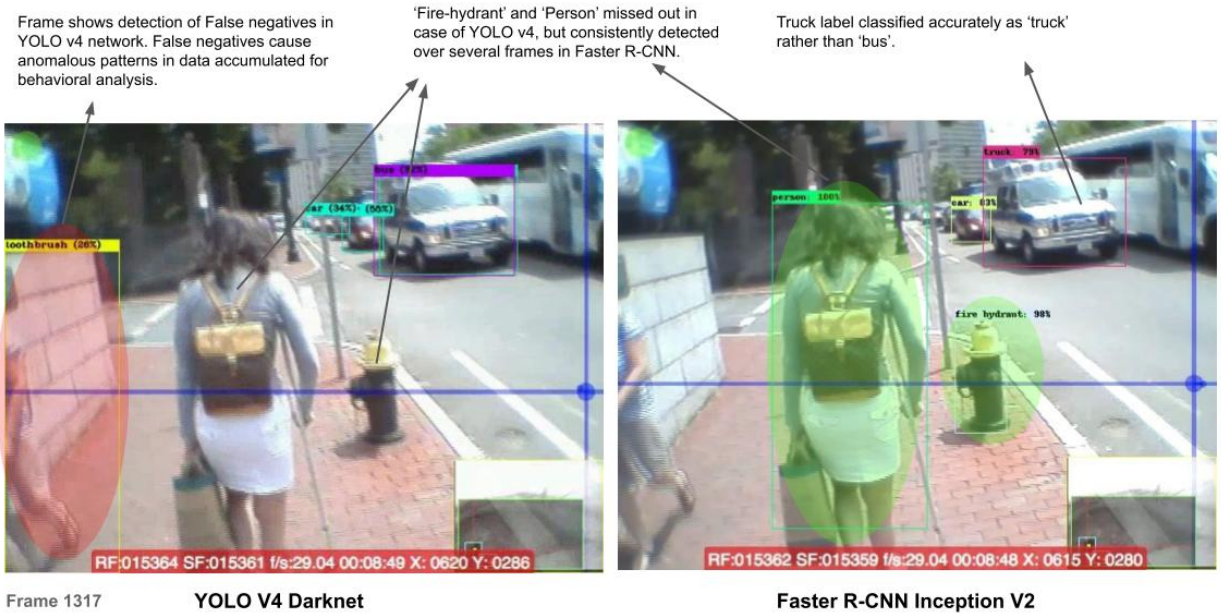

Fig. 6. Frame-wise comparison of Faster R-CNN network with YOLOv4 Darknet

- A tuple of 4 values normalized between [0,1] contain the coordinates of the bounding box namely (center x-coordinate, center y-coordinate, width, height) in the format (x, y, w, h) where x and y are conditioned depending on the cell location.
- Each cell is associated with its individual confidence score which indicates the likelihood of the presence of object. Pr(containing an object) x IoU(pred, truth) . Pr: Probability, IoU: Intersection over Union.
- The probability of an object contained by a cell belonging to every class $C_i, i = 1 \ldots K$ is predicted by Pr(the object belongs to the class $C_i \mid$ containing an object). This allows the model to predict only one set of class probabilities per cell regardless of the number of bounding boxes, B.
- Cumulatively, an image contains $NxNxB$ bounding boxes where each box corresponds to 4 location predictions, 1 confidence score, and K conditional probabilities for object classification.
- Finally, the final layer of YOLO's CNN Network outputs a tensor of size $NxNx(5B + K)$

A.2. Faster R-CNN

R-CNN, coined by [10], is a Convolutional Neural Network (CNN) with an added region-proposal algorithm which hypothesizes object locations. This network employs a selective search to initially extract a fixed number of regions (2000). Further, using a greedy algorith, this network merges similar regions together to obtain the selected regions where object detection is applied. Since R-CNN came with a speed bottleneck, both in terms of training and testing, the authors designed an enhanced algorithm entitled Fast R-CNN [11] using a shared convolutional feature map that the convolutional neural network would generated from the input image which is used to extract the Regions of Interest (RoI). While Fast R-CNN was arguably better

in terms of both training and testing time, the improvement was not dramatic because the region proposals were generated separately by another model which tends to be expensive. However, Ren et al. [12] proposed a Faster R-CNN algorithm which introduced the Region Proposal Network (RPN) by integrating the region proposal algorithm in the CNN model itself. Faster-RCNN introduces the construction of an ensemble of a unified model composed of RPN and Fast R-CNN with shared convolutional feature layers which is trained end-to-end to predict both- the object bounding boxes and objectness scores in a computationally inexpensive manner (10ms per frame).

Workflow explaining the Faster R-CNN architecture in 8:

- The RPN (region proposal network) is fine-tuned end-to-end for the region proposal task, further initialized by the pre-train image classifier. The positive samples are assigned an IoU score > 0.7 and negative samples an IoU score < 0.3 which might be indirectly treated as a threshold.
- A spatial window cover of size `n x n` is slid over the convolutional feature map of the frame.
- At the center of each sliding window, multiple regions of varying scales and ratios are predicted simultaneously. An anchor is a combination of `(sliding window center, scale, ratio)`. For example, 3 scales + 3 ratios => k=9 anchors at each sliding position.
- Finally, a 'Fast R-CNN' object detection model is trained using the proposals generated by the RPN.
- The Fast R-CNN network is used to then initialize RPN training. The Shared Convolutional layers are kept and simultaneously, the RPN-specific layers are fine-tuned. Now, the detection network and RPN have shared convolutional layers which completes the workflow.

Loss function for Faster R-CNN network: The loss function of the Faster R-CNN network can be described in the following manner-

The multi-task loss function combines the losses of classification and bounding box regression:

$$\mathscr{L} = \mathscr{L}_{\text{cls}} + \mathscr{L}_{\text{box}}$$

$$\mathscr{L}\left(\{p_i\},\{t_i\}\right) = \frac{1}{N_{\text{cls}}}\sum_i \mathscr{L}_{\text{cls}}\left(p_i, p_i^*\right) + \frac{\lambda}{N_{\text{box}}}\sum_i p_i^* \cdot L_1^{\text{smooth}}\left(t_i - t_i^*\right)$$

where $\mathscr{L}_{\text{cls}}$ is the log loss function over two classes, as we can easily translate a multi-class classification into a binary classification by predicting a sample being a target object versus not. L_1^{smooth} is the smooth L1 loss.

$$\mathscr{L}_{\text{cls}}\left(p_i, p_i^*\right) = -p_i^* \log p_i - \left(1 - p_i^*\right)\log\left(1 - p_i\right)$$

The variables used to defined the loss function can be enlisted in the form: $|p_i|$ Predicted probability of anchor i being an object. $\| p_i^*$ | Ground truth label (binary) of whether anchor i is an object. $\|t_i$ | Predicted four parameterized coordinates. $\|t_i^*$ | Ground truth coordinates. | | N_{cls} | Normalization term, set to be mini-batch size (256) in the paper. | N_{box} | Normalization term, set to the number of anchor locations (2400) in the paper. $\| \lambda$ | A balancing parameter, set to be ~ 10 in the paper (so that both $\mathscr{L}_{\text{cls}}$ and $\mathscr{L}_{\text{box}}$ terms are roughly equally weighted).

B. Comparison of Both Frameworks through Standard Metrics

Since YOLOv4 performance widely varies as per context and the dataset, we subjectively carried out our evaluation by localizing the detected objects in each frame, penalizing the performance for each undesired object detected or object missed out with separate categories for occlusion based missing or inconsistent frame tracking. Table 1 proposes a direct comparison between YOLOv4 Darknet and Faster R-CNN based on theoretical information on the architecture of the given models.

Quantifying rate of False-positives and False-negatives in both networks Amar et al. [13] conducted a comparative study between the frameworks (Faster R-CNN Inception V2, Faster R-CNN Resnet 50, YOLO V3 and YOLO V4) and found that the difference is False positives, negatives and precision is closely intertwined with the dataset chosen for the study. While the above research was conducted on two aerial datasets, the performance of both models widely varied with the nature of the dataset- (Stanford Dataset and Prince Sultan University (PSU) Dataset). The representation of the images and nature of images widely influenced the performance of the models. The Stanford dataset had nearly 30 times as many object instances to be trained on than PSU dataset per image. Following this, one of the conclusions made by the authors were that the YOLO V4 was specifically tuned for the COCO dataset and did not yield as high Average Precision values for the rest of the datasets. It could also be subjectively observed that YOLO V4 had a higher tendency of detecting fainter, distant objects which Faster R-CNN could not easily catch, and while this hints at YOLO V4 being a better algorithm for this niche of datasets, these results cannot be smoothly extrapolated in our case. The reason is that Faster R-CNN performs better than YOLO V4 for occluded images considering the nature of the algorithm; and with most of the dataset consisting of occlusions, the result changes drastically.

4. ALGORITHM

We used the Faster R-CNN Inception V2 Framework to carry out the object detection for the Focal view of the gaze combined with gaze estimation done discretely and later combined into a single analysis algorithm to calculate the description of different objects being gazed which breaks down into qualitative and quantitative analysis. The results section is also further broken down into two parts:

- Raw data analysis
- Visual description of quantitative results

The eye-tracking data split into yaw, pitch and roll per frame is processed after factoring in the results from IMU sensors and the cumulative data is quantified in the form of X and $\mathbf{Y}$ axis spatial locations of eye-gaze vectors. These spatial locations for each eye-gaze vector is associated with N other object entities detected in the given frame corresponding to the gaze. For e.g. each frame may have x_a *person* objects or x_b *car* objects, while the gaze vector might only be focusing on one object or on none of them. Our approach tries to calculate the most-probable results for the object being gazed and encapsulates these results in the form of spatio-temporal graphs which capture the spatial location for each object and the gaze moving through time.

The first step was to analyze the Euclidean Distance between the eye-gaze vector and the centroid of the bounding box of detected

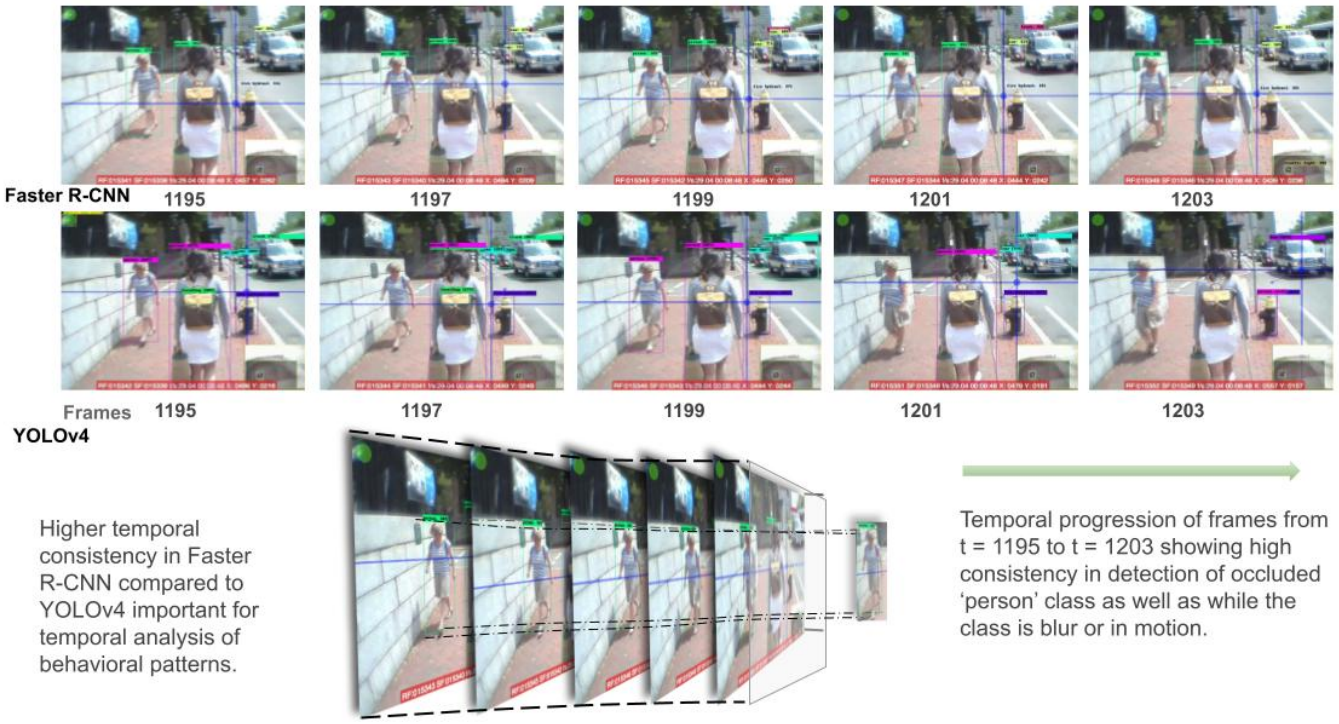

Fig. 7. Consistency analysis of both networks - YOLOv4 Darknet and Faster R-CNN Inception v2

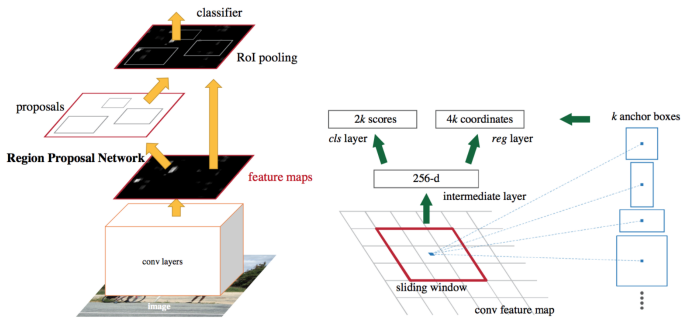

Fig. 8. Network architecture for Faster-RCNN model

objects. This analysis pointed out the magnitude of difference between the Euclidean Distances between objects which were gazed and objects not gazed.

There are three conditional stages employed to qualitatively assess whether an object was gazed by the eye.

.1. point-in-polygon (PIP) algorithm to determine if the spatial coordinate of eye-gaze lies within the bounding box.

The point-in-polygon (PIP) method is the most optimum method to compute the relative position of eye-gaze coordinate when applied to bounding-boxes as compared to the geofencing algorithm. What simplifies even more here is that the structure of bounding box, in the form of rectangles rather than polygons, simplifies the computational method to determine whether the point lies within the box, thus decreasing computation time.

Algorithm 1. Point-in-Rectangle (PIR) Algorithm

procedure POINTINRECTANGLE($point, rectangle$)
  $x, y \leftarrow point$
  $x_1, y_1, x_2, y_2 \leftarrow rectangle$
  if $(x, y) \in \{(x_1, y_1), (x_1, y_2), (x_2, y_1), (x_2, y_2)\}$ **then**
    return True
  if $x_1 < x < x_2$ & $y_1 < y < y_2$ **then**
    return True
  return False

.2. Euclidean Perimeter Sum point-in-polygon (PIP) Algorithm to determine if the spatial eye-gaze coordinate lies within the bounding box for a cluster of points.

Despite its increased computational demands, this method has the benefit of expediting the process of assigning a point to a bounding box when the vertices are dispersed. The case-specific modification of the algorithm is particularly beneficial in instances where bounding boxes have not undergone pre-processing and are in an unrefined state. This approach was initially employed to compute a set of eye-gaze coordinates for a smaller video frame dataset before transitioning to the first method. It is noteworthy that the case-specific modification of this algorithm may be advantageous in comparable situations where time can be conserved through the direct assignment of a point to a bounding box without the necessity of consulting data tables.

.3. Confirmatory Thresholding approach over a Euclidean algorithm to verify poin-in-polygon (PIP) classifications

There are several circumstances in which *point-in-polygon (PIP)* and *Euclidean Perimeter Sum PIP* algorithms fail to correctly identify eye-gazes that are targeted at objects. One such scenario occurs when attempting to detect oblong objects, such as pillars, road signs, and utility poles, which are frequently the focus of human gaze but may not be detected due to the shape of their bounding boxes. Bounding

	YOLOv4	Faster R-CNN
Phases	Concurrent bounding box regression, and classification.	RPN + Fast R-CNN object detector.
Neural network type	Fully convolutional.	Fully convolutional (RPN and 4 detection network).
Backbone feature extractor	CSPDarknet53 (53 convolutional layers).	VGG-16 or Zeiler Fergus(ZF). Other feature extractors can also be incorporated.
Location detection	Anchor-based	Anchor-based
Number of anchor boxes	Using multiple anchors for a single ground truth	3 scales and 3 aspect ratios, yielding k = 9 anchors at each sliding position.
Default Anchor sizes	(12,16), (19,36), (40,28), (36,75), (76,55), (72,146), (142,110), (192,243), (459,401)	Scales: (128,128), (256,256), (512,512). Aspect ratios: 1:1, 1:2, 2:1.
IoU thresholds	One (at 0.213)	Two (at 0.3 and 0.7).
Loss function	Complete IoU loss: CIoU	Multi-task loss: - Log loss for classification. - Smooth L1 for regression.
Input size	Different possible input sizes (n × n with n multiple of 32).	- Conserves the aspect ratio of the original image. - Either the smallest dimension is 600, or the largest dimension is 1024.
Batch size	Default value: 64.	Default value: 1

Algorithm	Feature Extractor	Input Size	AP	TP	FN	FP	Precision	Recall	F1 Score	FPS	Inference Time (ms)
Faster R-CNN	Inception v2	992 × 550 (variable)	0.739	548	190	11	0.980	0.743	0.845	9.5	105
Faster R-CNN	Inception v2	608 × 608 (fixed)	0.731	541	197	14	0.975	0.733	0.837	9.5	105
YOLOv4	CSPDarknet-53	320 × 320 (fixed)	0.961	715	23	59	0.924	0.969	0.946	22.4	45
YOLOv4	CSPDarknet-53	416 × 416 (fixed)	0.965	720	18	66	0.916	0.976	0.945	19.4	52
YOLOv4	CSPDarknet-53	608 × 608 (fixed)	0.950	715	23	66	0.915	0.969	0.941	13	77

Table 2. Numerical score for different metrics associated with PSU Dataset.

Algorithm	Feature Extractor	Input Size	AP	TP	FN	FP	Precision	Recall	F1 Score	FPS	Inference Time (ms)
Faster R-CNN	Inception v2	600 × 816 (variable)	0.202	1780	6351	1813	0.495	0.219	0.304	19.2	52
Faster R-CNN	Inception v2	608 × 608 (fixed)	0.317	2916	5215	2654	0.524	0.359	0.426	21.1	47
YOLOv4	CSPDarknet-53	320 × 320 (fixed)	0.157	1278	6853	5	0.996	0.157	0.272	21.1	47
YOLOv4	CSPDarknet-53	416 × 416 (fixed)	0.202	1646	6485	1	0.999	0.202	0.337	18.5	54
YOLOv4	CSPDarknet-53	608 × 608 (fixed)	0.209	1701	6430	64	0.964	0.209	0.344	12.5	80

Table 3. Numerical score for different metrics associated with Stanford Dataset

boxes that are not square in shape may have a lower probability of accurately detecting eye-gazes due to the configuration of their shape

Algorithm 2. Euclidean Point-in-Polygon (PIP) Algorithm

procedure POINTINPOLYGON($\mathbf{p}, \mathbf{P}$)
  $\mathbf{p} = (x, y)$
  $\delta = 0$
  $n = |\mathbf{P}|$
  for $i \leftarrow 1$ **to** n **do**
    $\mathbf{P}_i = (x_1, y_1, x_2, y_2)$ ▷ calculate the distance between the point and the current side
    $\delta \leftarrow \delta + \sqrt{(x - x_1)^2 + (y - y_1)^2} + \sqrt{(x - x_2)^2 + (y - y_2)^2}$
  $L = 0$
  for $i \leftarrow 1$ **to** n **do**
    $\mathbf{P}_i = (x_1, y_1, x_2, y_2)$
    $L \leftarrow L + \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$
  if $\delta = L$ **then**
    return True
  return False

and the area they enclose. Additionally, the bounding boxes of temporally moving objects may fluctuate significantly, causing eye-gazes to be erroneously detected as being within the bounding box when they are not. Moreover, in cases where the area encompassed by a bounding box is significantly larger than the area occupied by the object, the eye-gaze may lie within the void of the bounding box with no intention of fixating on the object. These are just a few examples of the limitations that restrict the autonomous implementation of the first and second methods.

To address these issues, a confirmatory third approach has been developed that employs thresholding by averaging sample Euclidean distances between the eye-gaze coordinates and the centroid of the bounding box for cases returned as "True" in the first two methods. A second confirmatory threshold is applied to determine the diagonal distance between the radially opposite vertices of the bounding boxes. This confirmatory threshold is used for objects that are oblong in size and it is difficult to determine whether an object is being gazed at solely based on the Euclidean threshold from the centroid. This test eliminates all objects for which the eye-gaze is greater than the diagonal distance from the centroid to one of the vertices. This is because oblong objects have extremely large diagonal distances from the centroid to the vertices, and if the Euclidean distance surpasses that distance, it is highly unlikely that the object would be gazed.

The algorithmic proof of the algorithm is given below:

Algorithm 3. Euclidean Distance Threshold Algorithm

procedure CHECKDISTANCE($\mathbf{e}, \mathbf{B}, \mu$)
  $\mathbf{e} = (x_e, y_e)$
  $\mathbf{B} = (x_1, y_1, x_2, y_2)$
  $c_x \leftarrow \frac{x_1 + x_2}{2}$
  $c_y \leftarrow \frac{y_1 + y_2}{2}$
  $d \leftarrow \sqrt{(x_e - c_x)^2 + (y_e - c_y)^2}$
  if $d > \mu$ **then**
    $\delta \leftarrow \sqrt{(c_x - x_1)^2 + (c_y - y_1)^2}$
    if $d > \delta$ **then**
      return True
  return False

Here, μ is the threshold calculated as the sum of Mean of data points and the mean of the standard deviation of the data points from the mean. Algebraically, given as:

$$\mu = \frac{\sum_{i=1}^{n} \sigma_i}{n} + \frac{\sum_{i=1}^{n} x_i}{n}$$

Where σ_i is the standard deviation of the i^{th} set of data.

$$\sigma_i = \sqrt{\frac{\sum_{i=1}^{n} (x_i - \mu_x)^2}{n - 1}}$$

Where x_i is the i^{th} element of the data set, μ_x is the mean of the data set, and n is the total number of elements in the data set.

5. RESULTS AND DATA ANALYSIS

Based on the previous Algorithms to classify whether an object was gazed or not, the results for the following research are tabulated and then the data is sequentially visualized to identify the cognitive patterns in patients with Homonymous Hemaniopia.

The dataset analyzed in this research comprises 32 minutes of video input obtained from a patient diagnosed with Homonymous Hemianopia, captured at 29 frames per second. The dataset yields a total of 56.6K frames visualized and 431.6K objects detected across the frames, providing a rich source of data for analysis.

A more granular examination was conducted on a subset of frames, specifically frames 16581 to 18199, which were chosen as they provide a representative sample of the patient's eye-gaze patterns while traversing a road, particularly at a pedestrian crossing. This subset of frames presents an equal distribution of vehicles and pedestrians, providing valuable insights into the patient's gaze patterns in complex interactions with the surroundings, specifically in cases where multiple objects are in motion simultaneously, resulting in multiple points of attention, contextually within the patient's visual field. To better evaluate the 3D visualizations generated for eye-gaze spatio-temporal features over time, we divide the Frames from 16581 to 18199 into groups of three. The first one containing Frames 16581 to 17009 i.e 571 to 586 seconds, Frames 17009 to 17430 i.e 586 to 601s and Frames 17430 to 18199 i.e 601 to 627s of the encoded video frame.

A. Visualizing Eye Gaze Distribution over Focal View

A.1. Kernel Density Estimation Plots

An important aspect of studying Homonymous Hemaniopia in patients is analyzing the distribution of Eye Gaze spatially. This is an important visualization specifically for the selected subset of frames during Street Crossing. We visualized this Spatio-Temporal data points through a Kernel Distribution Estimation Plot which depicts the probability density function of the non-parametric data points. Fig. 9 represents the KDE plots for eye gaze detections over objects utilizing the PIP algorithm while Fig. 10 represents the KDE plots for eye gaze detections over objects through the Euclidean Thresholding algorithm. The visualizations present more detail by distinguishing between the heatmap for *Objects which are gazed* by *Objects which are not gazed*. Clearly, the heatmap for objects which are not gazed is larger than the heatmap for objects which aren't and for the classes which are more abundant throughout the frames, the heatmap is centred at the centroid of field of view.

The table 5 quantifies the distribution between objects gazed using the Point-in-Polygon algorithm 1 and the Euclidean Thresholding Algorithm 3 and it was found that the objects gazed using the Thresholding algorithm were greater than the PIP algorithm by a significant amount. Multiple objects rapidly scanned by the human-eye through peripheral vision are not completely calculated using the PIP approach and this is further elaborated in Subsection 6. While

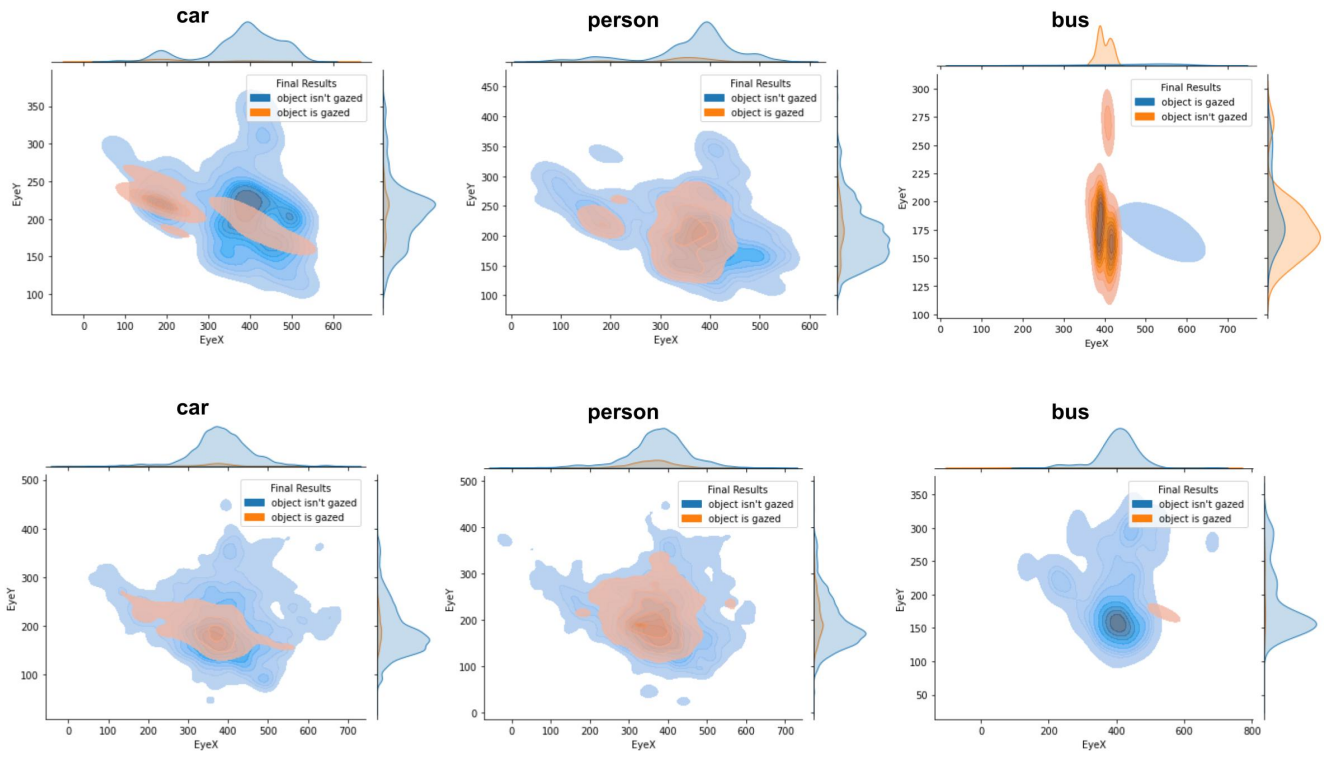

Fig. 9. Point-in-Polygon (PIP) implementation of a Kernel Distribution Plot (KDE) of eye-gaze data visualized during street-crossing between frames 16581 to 18199 (top) and entire dataset (bottom)

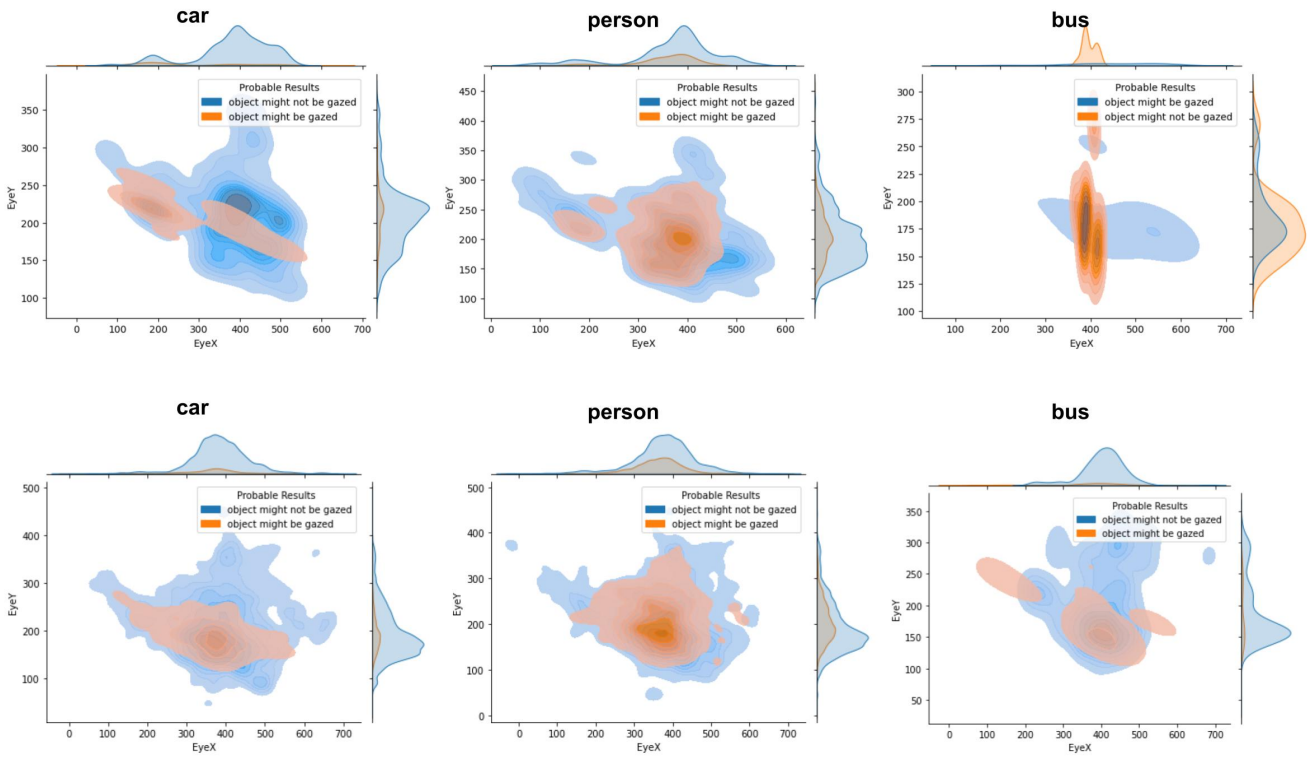

Fig. 10. Euclidean Thresholding Algorithm implementation of a Kernel Distribution Plot (KDE) of eye-gaze data visualized during street-crossing between frames 16581 to 18199 (top) and entire dataset (bottom)

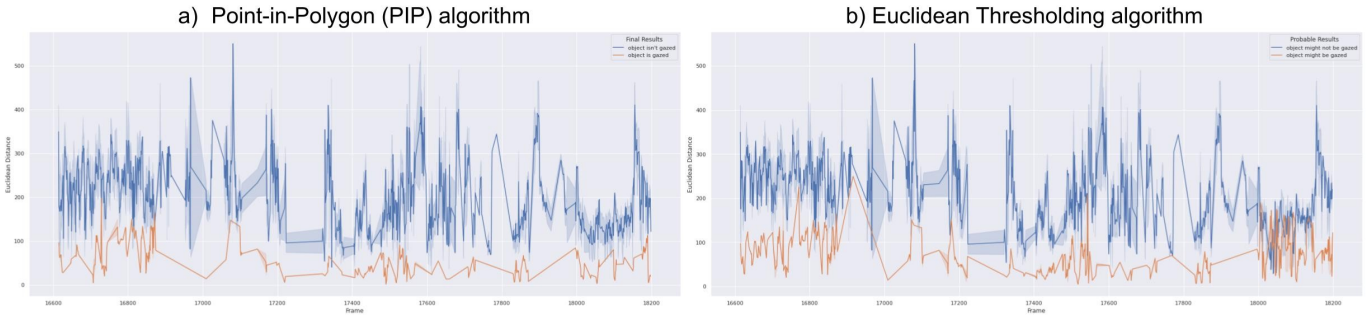

Fig. 11. Comparing Euclidean distance between objects which are gazed and which aren't through PIP algorithm and Euclidean Thresholding algorithm

	Point-in-Polygon (PIP) algorithm	Euclidean Thresholding Algorithm	Total Detected objects
n(car)	235	284	3060
n(person)	1127	2317	10924
n(traffic light)	0	0	59
n(motorcycle)	0	0	28
n(fire hydrant)	0	0	239
n(bus)	23	30	91

Table 4. Quanitfying increment in number of objects gazed through the Euclidean Thresholding Algorithm over the street crossing dataset

	Point-in-Polygon (PIP) algorithm	Euclidean Thresholding Algorithm	Total Detected objects
n(car)	4672	8018	70698
n(person)	17866	31216	118611
n(traffic light)	26	33	5091
n(motorcycle)	49	61	447
n(fire hydrant)	29	65	854
n(bus)	50	118	1258

Table 5. Quanitfying increment in number of objects gazed through the Euclidean Thresholding Algorithm over the entire dataset

Saliency Detection is for further discussion, in a nutshell they isolate pixels within an image which correlate to the maximum neural activation and are likely to be gazed first.

Compared to the PIP approach, there is an overall 52.43% increase in detections through Euclidean Thresholding algorithm over street crossing frames. Objects which were gazed through peripheral view or through other possible limitations discussed in 4 through the PIP algorithm is covered in the Euclidean approach. We further investigate the deviation of the gaze in a Homonymous Hemaniopia patient from more salient regions for a normal eye using ConvLSTM's in 6.

A.2. Temporal Gaze Graph representing rapid/abrupt or slow/steady gaze moments over a period of Frames in the Field of view and the tendency of gaze to shift towards higher density object clusters

After analyzing Kernel Density Estimation (KDE) plots, we visualized Spatio-temporal 3D plots of movement of objects in the frames over time. In the same plot, the variance of eye gaze was analyzed by extracting X and Y coordinates from Yaw, Pitch and Roll data combined from head and eye movements.

A.3. Complexity of fixation determination through Hurst's exponent.

To gauge the intricacy of time series for fixation duration, we employ the MF-DFA approach. First introduced by Kantelhardt et al.[14], **??**, this technique has been utilized in various time series such as sunspot activity, traffic data [15], economic data [16], heart rate measurements [17], brain electrical signals in humans , streamflow records, and wind data.

Multifractal Detrended Fluctuation Analysis (MF-DFA) is a mathematical method used to analyze the scaling behavior of complex signals, including eye gaze data. The method is used to identify the presence of multiple fractal scaling exponents in the data, which can indicate the presence of multiple patterns or structures in the data.

In the context of eye gaze tracking, MF-DFA can be used to analyze the scaling behavior of eye movements and fixations, which can provide information about the gaze stability, cognitive load, and attention allocation during a visual task [18].

The procedure of MF-DFA is as follows. Consider the time series $x(i)$ of length N, its cumulative series is defined as

$$Y(i) = \sum_{k=1}^{i} \big(x_k - \langle x \rangle\big); \quad i = 1, 2, \cdots N,$$

The average of the time series over the entire data range is represented by $\langle x \rangle$. The values of $Y(i)$ can be considered as steps in a random walk, where each step represents the fluctuation of the time series around its average at a specific time. The next step involves removing the local trends from the profile series Y. This is done by dividing the $Y(i)$ series into segments of length dl and using a least-square fit to fit a polynomial Y_v to the data in each segment. The variance of the data in each segment around the fitted polynomial is expressed as:

$$F^2(dl, v) = \frac{1}{s} \sum_{i=1}^{s} \{Y[(v-1)dl + i] - Y_v(i)\}^2; v = 1, 2, \cdots N_s$$

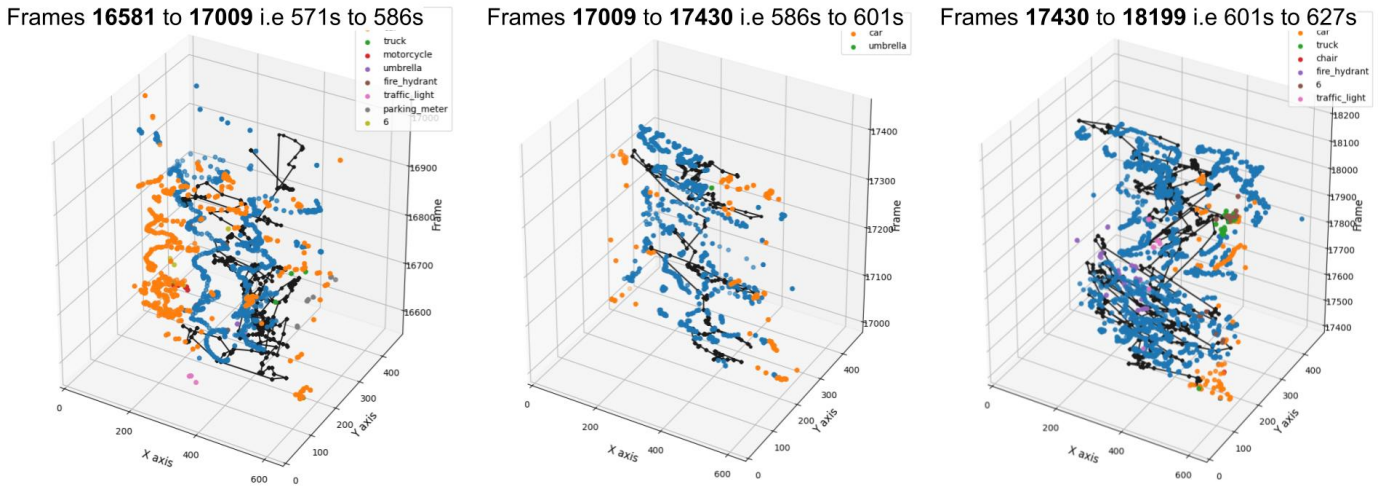

Fig. 12. Frames for street crossing visualized with all objects in the field of view

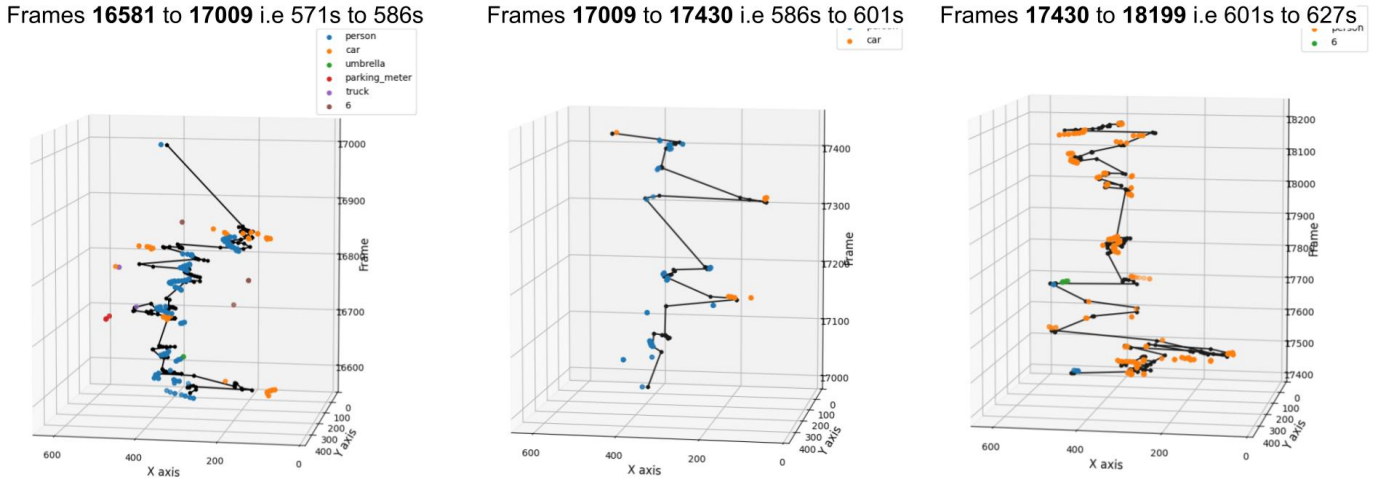

Fig. 13. Frames for street crossing visualized with onl gazed objects in the field of view

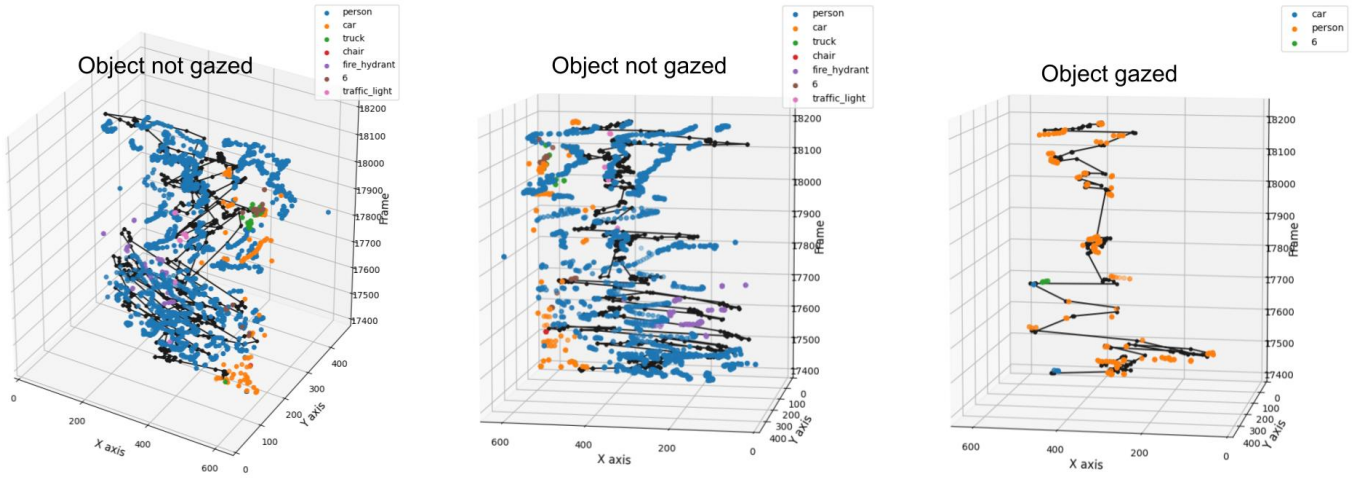

Fig. 14. Frames 17430 to 18199 i.e 601s to 627s

Fig. 15. 3D visualization of spatio-temporal change in eye-gaze location in the x-y plane over frames.

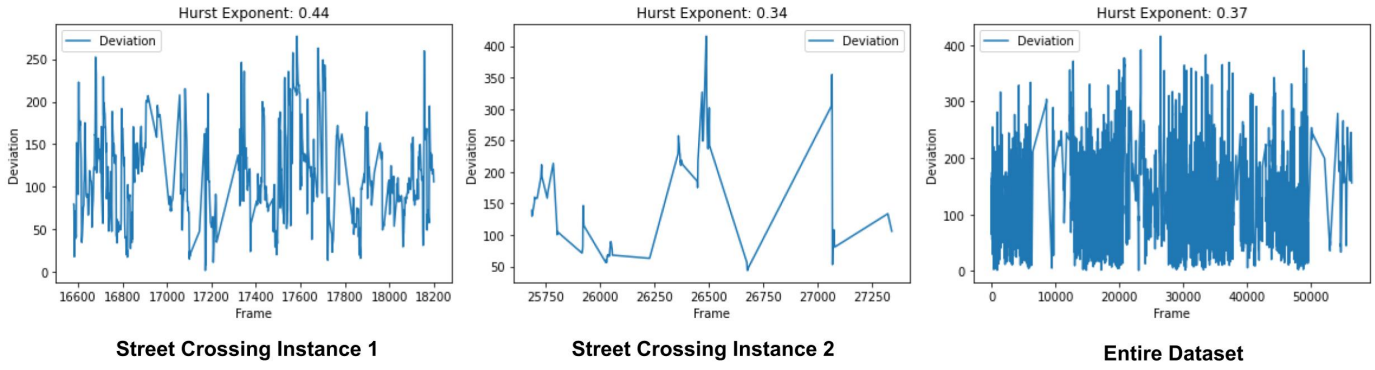

Fig. 16. Calculation of the Hurst's exponent for the deviation of eye-gaze from the mean position for three instances of dataset analyzed. Deviation is in the units of Euclidean Distance from the centroid of the Focal View.

The value of N_s is calculated by dividing the total number of data points, N, by the size of each data segment, dl. In this study, linear polynomials, specifically first order DFA or DFA1, are used for detrending the data. Once the detrending is complete, the multifractal properties of the time series are analyzed. This involves defining the q-th fluctuation moment for any non-zero real value of q.

$$F_q(dl) = \left(\frac{1}{N_s} \sum_{v=1}^{N_s} \left[F^2(dl, v) \right]^{q/2} \right)^{1/q}.$$

For $q = 0, F_0(dl)$ can be written as

$$F_0(dl) = \exp\left(\frac{1}{2N_s} \sum_{v=1}^{N_s} \ln\left(F^2(dl, v) \right) \right).$$

For large dl, the q-th fluctuation moment, $F_q(dl)$, follows a scaling law that states that $F_q(dl)$ is proportional to $s^{h(q)}$, where $h(q)$ is known as the generalized Hurst exponent. A monofractal time series has similar statistical fluctuations at different scales, making $h(q)$ independent of q. When $q = 2$, $h(2)$ is the standard Hurst exponent, H. The standard Hurst exponent for uncorrelated time series is $H = 1/2$, for anti-correlated it is $H < 1/2$, and for correlated it is $H > 1/2$. [19]

Algorithm 4. Calculating Hurst's Exponent

1: **Create** the range of lag values: $lags \leftarrow \text{range}(2, 100)$
2: **Calculate** the array of the variances of the lagged differences:

$$\tau \leftarrow [\sqrt{(\text{std}(ts[lag:] - ts[:-lag]))} \text{ for } lag \text{ in } lags]$$

3: **Use** a linear fit to estimate the Hurst Exponent:

$$\varphi \leftarrow \text{polyfit}(\log(lags), \log(\tau), 1)$$

4: **Return** the Hurst exponent **H** from the polyfit output:

$$\mathbf{H} \leftarrow \varphi[0] \cdot 2$$

A Hurst exponent of 0.5 indicates random gaze tendency, while values greater than 0.5 indicate positive persistence, meaning that large deviations are followed by larger deviations and vice versa. On the other hand, values less than 0.5 indicate negative persistence, meaning that large deviations are followed by smaller deviations and vice versa.

In our case, a calculated Hurst exponent of 0.3 to 0.45 indicates that the fluctuations in the eye gaze deviations persist over time, but not to a large extent. This suggests that the fluctuations in eye gaze deviations are not random, but not as persistent as those with a Hurst exponent close to 0.5 or higher.

Fig. 17. Results using a Saliency Attentive ConvLSTM model on an instance of street crossing

6. SALIENCY DETECTION

The human visual system relies on the mechanism of visual attention to effectively navigate the vast and intricate visual landscape of the world. This is achieved by selectively focusing on the most salient or pertinent elements within the scene. In the field of computer vision, this process is replicated through the use of saliency detection, which allows for the computational identification of regions within an image that are likely to capture the attention of human observers.

Studies in the field of visual cognition have revealed that when humans are looking at an image without a specific task in mind, their gaze does not evenly cover every part of the picture. Instead, attention mechanisms direct their gaze towards parts of the image that are important and noticeable [20]. A lot of research has been dedicated to replicating this selective visual behavior through computational saliency, which can be utilized in various applications such as image retargeting [21], object recognition [22], video compression [23], tracking [24], and image captioning [25] which rely on data.

Perception of visual stimuli is not uniform across all creatures, including humans and primates. Within the visual field, there is a specific area called the fovea that has a much higher density of receptors in the retina, allowing for clearer vision. To gain more in-depth information about the visual world, it is necessary to make eye movements to direct the fovea towards relevant or intriguing objects. These movements often include fixations, where the gaze remains still on a specific point, and rapid shifts called

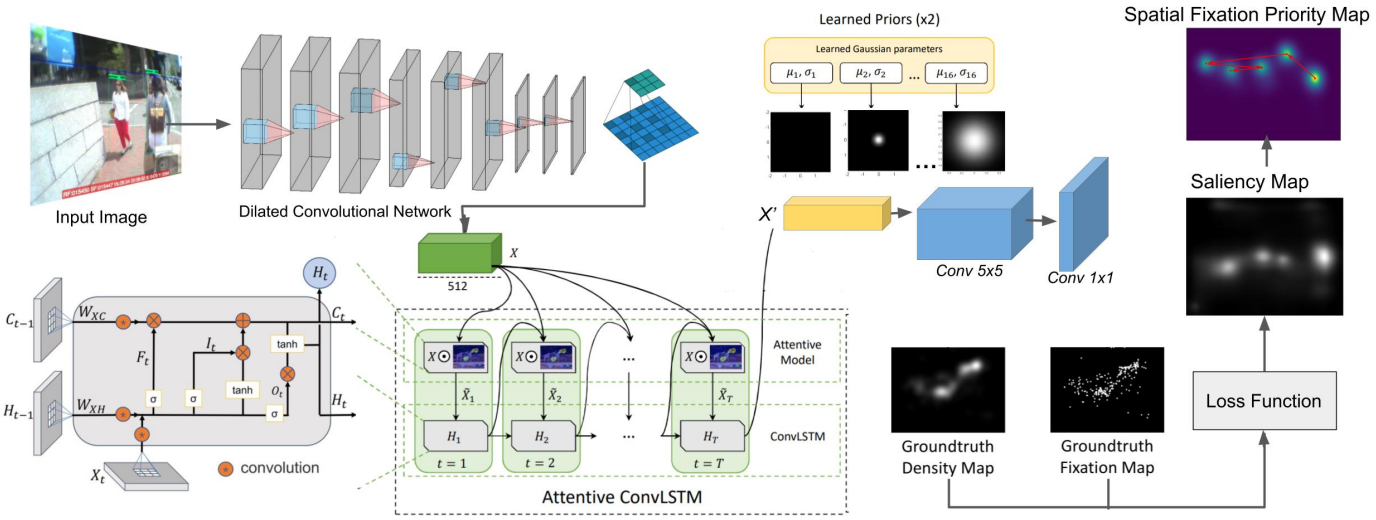

Fig. 18. An overview of the modified Saliency Attentive Model to incorporate order of spatial fixation locations is given. The input image is processed through a Dilated Convolutional Network to compute a set of feature maps. An Attentive Convolutional LSTM enhances the saliency features sequentially using an attentive recurrent mechanism. The predictions are combined with multiple learned priors to capture the tendency of humans to focus on the center of the image. During training, the network will be encouraged to minimize a combination of different loss functions to account for various quality aspects that the predictions should possess.

saccades, where the gaze moves from one fixation to another. While studying visual patterns in patients with Homonymous Hemianopia, identifying the deviation of gaze from the most salient objects in the visual frame is crucial to understanding the receptive behavior of such patients. This gains further importance in Street Crossing instances where directing the fovea towards objects moving with high velocity such as cars, trucks or rapidly moving pedestrians becomes important when low level features from peripheral vision aren't enough to make split-second decisions while crossing the road.

A. Model Architecture

In this research, we utilize an Attentive Convolutional Long Short-Term Memory network (Attentive ConvLSTM) that iteratively focuses on relevant spatial locations to refine saliency features to quantify the most salient regions within the Field of View of the patient. Here, we develop upon the existent Saliency Attentive Model (SAM) by Cornia et al. [26] by introducing another algorithm to analyze for the priority of the Spatial Fixation Layers within the Saliency Map through by weighing each point of saliency using a threshold.

This method uses the ResNet model [27] to get features from the image. This is achieved through a deep spatial contextual LSTM to look at the image horizontally and vertically and combine global and local information to determine the important parts of the image. A significant body of research has explored the concept of illuminating neural model activations through backpropagation-based techniques [28]. It is crucial to recognize that this area of inquiry differs fundamentally from the field of saliency prediction as its objective is not to imitate human gaze patterns.

The key innovation of the Saliency Attention Model [26] is the utilization of an Attentive Convolutional model, which processes saliency features recurrently by focusing on different regions of a tensor. This represents a novel use of LSTM, which usually relies on the notion of time. The predictions are then combined with multiple learned priors that model the human gaze center bias. The authors employ a Convolutional Neural Network to extract feature maps from input images and use a Dilated Convolutional Network instead of a pre-defined CNN to alleviate the rescaling effects that can impair saliency prediction performance. The network is trained using a combination of loss functions that take into account multiple quality aspects simultaneously.

A.1. Attentive ConvLSTM model

Long Short-Term Memory networks [29] have demonstrated good performance on several tasks that involve time dependencies [30], [31], [32]. However, they cannot be directly used for saliency prediction, as they operate on sequences of time-varying vectors.To overcome this, traditional LSTM models can be extended to work on spatial features by substituting dot products with convolutional operations in the LSTM equations. Additionally, they exploit the sequential nature of LSTM to process features iteratively, rather than using the model to handle temporal dependencies in the input.

The LSTM network takes in a stack of features derived from an input image X, then produces a more refined set of feature maps $X^{'}$ that are fed into a learned prior module. It operates by sequentially updating an internal state through three sigmoid gates, using specific equations.

$$
\begin{aligned}
I_t &= \sigma\left(W_i * \tilde{X}_t + U_i * H_{t-1} + b_i\right) \\
F_t &= \sigma\left(W_f * \tilde{X}_t + U_f * H_{t-1} + b_f\right) \\
O_t &= \sigma\left(W_o * \tilde{X}_t + U_o * H_{t-1} + b_o\right) \\
G_t &= \tanh\left(W_c * \tilde{X}_t + U_c * H_{t-1} + b_c\right) \\
C_t &= F_t \odot C_{t-1} + I_t \odot G_t \\
H_t &= O_t \odot \tanh\left(C_t\right).
\end{aligned}
$$

The gates, I_t, F_t, and O_t, the candidate memory, G_t, the memory cell, C_t and C_{t-1}, and the hidden state, H_t and H_{t-1}, are all 3-dimensional tensors with 512 channels each. The convolutional operator, represented by $*$, and all the parameters W, U, and b are 2-dimensional convolutional kernels and learned biases. The LSTM layer's input, $\tilde{X}t$, is calculated at each timestep through an attentive mechanism, which involves generating an attention map by convolving the previous hidden state, $Ht-1$, and the input, X, passing it

Fig. 19. Examples of Saliency maps predicted on the SERI Homonymous Hemaniopia Dataset using a modified Attentive ConvLSTM with learned priors

through a tanh activation function, and then convolving it with a one-channel convolutional kernel.

$$Z_t = V_a * \tanh\left(W_a * X + U_a * H_{t-1} + b_a\right).$$

The output of this operations is a 2-d map from which we can compute a normalized spatial attention map through the softmax operator:

$$A_t^{ij} = p\left(att_{ij} \mid X, H_{t-1}\right) = \frac{\exp\left(Z_t^{ij}\right)}{\sum_i \sum_j \exp\left(Z_t^{ij}\right)}$$

where A_t^{ij} is the element of the attention map in position (i, j). The attention map is applied to the input X with an elementwise product between each channel of the feature maps and the attention map:

$$\tilde{X}_t = A_t \odot X.$$

Psychological studies have revealed that observers' gazes tend to be biased toward the center of images. This is mainly due to the tendency of photographers to place objects of interest at the center, as well as the expectation of finding the most informative content around the center as a result of repeated viewing of such images. Another factor contributing to this behavior is the interestingness of the scene. In the absence of highly salient regions, humans tend to look at the center of the image.

Based on this research, the inclusion of center priors is a crucial element in several recent works on saliency prediction [33], [34], [35]. Saliency Attentive Model allows the network to learn its own priors. To reduce the number of parameters and simplify the learning process, the priors are constrained to be 2-dimensional Gaussian functions, with the mean and covariance matrix being freely learnable. This approach enables the network to learn its own priors solely from data, without relying on assumptions from biological studies.

The center bias can be modelled by means of a set of Gaussian functions with diagonal covariance matrix. This is learned according to the following equation:

$$f(x, y) = \frac{1}{2\pi\sigma_x\sigma_y} \exp\left(-\left(\frac{(x-\mu_x)^2}{2\sigma_x^2} + \frac{(y-\mu_y)^2}{2\sigma_y^2}\right)\right).$$

The Saliency Attentive Model learns the parameters of 16 Gaussian functions through the course of its experiments and generates the relevant prior maps by combining the learned maps with a tensor that possesses 512 channels, resulting in a tensor with 528 channels. This tensor is then processed through a convolutional layer that features 512 filters, adding non-linearity to the model and proving to be more effective than previous works as demonstrated in section V-C. The entire prior learning component is repeated twice within the Saliency Attentive Model.

A.2. Loss Function

In order to account for various quality factors, saliency predictions are usually measured using different metrics. Based on this evaluation process, the Saliency Attentive Model introduces a new loss function that is a linear combination of three separate saliency evaluation metrics. The overall loss function is defined as follows:

$$L\left(\tilde{\mathbf{y}}, \mathbf{y}^{den}, \mathbf{y}^{fix}\right) = \alpha L_1\left(\tilde{\mathbf{y}}, \mathbf{y}^{fix}\right) + \beta L_2\left(\tilde{\mathbf{y}}, \mathbf{y}^{den}\right) + \gamma L_3\left(\tilde{\mathbf{y}}, \mathbf{y}^{den}\right)$$

The loss function of the Saliency Attentive Model is a linear combination of three different evaluation metrics for saliency prediction: Normalized Scanpath Saliency (NSS), Linear Correlation Coefficient (CC), and Kullback-Leibler Divergence (KL-Div). These metrics are commonly used to assess the performance of saliency prediction models. The predicted saliency map, $\tilde{\mathbf{y}}$, is compared to the groundtruth density distribution, $\mathbf{y}^{\text{den}}$, and the groundtruth binary fixation map, $\mathbf{y}^{fix}$, to calculate the three loss functions, with scalars α, β, and γ adjusting the balance between them. The NSS metric was specifically designed for the evaluation of saliency models and calculates the saliency map values at the eye fixation locations, normalized by the saliency map variance.

$$L_1\left(\tilde{\mathbf{y}}, \mathbf{y}^{fix}\right) = \frac{1}{N} \sum_i \frac{\bar{\mathbf{y}}_i - \mu(\bar{\mathbf{y}})}{\sigma(\bar{\mathbf{y}})} \cdot \mathbf{y}_i^{fix}$$

where i indexes the i^{th} pixel, $N = \sum_i \mathbf{y}_i^{fix}$ is the total number of fixated pixels and $\bar{\mathbf{y}}$ is normalized to have a zero mean and unit standard deviation.

The Linear Correlation coefficient or the Pearson's correlation coefficient treats the saliency and groundtruth density maps, $\bar{\mathbf{y}}$ and $\mathbf{y}_{den}$, as random variables measuring the linear relationship between them. The following equation explains the computation process:

$$L_2\left(\tilde{\mathbf{y}}, \mathbf{y}^{den}\right) = \frac{\sigma\left(\tilde{\mathbf{y}}, \mathbf{y}^{den}\right)}{\sigma(\bar{\mathbf{y}}) \cdot \sigma\left(\mathbf{y}^{den}\right)}$$

where $\sigma\left(\tilde{\mathbf{y}}, \mathbf{y}^{den}\right)$ is the covariance of $\tilde{\mathbf{y}}$ and $\mathbf{y}^{den}$.

The Kullback-Leibler Divergence (KL-Div) measures the amount of information lost when approximating the groundtruth density distribution with the predicted saliency distribution. This metric takes a

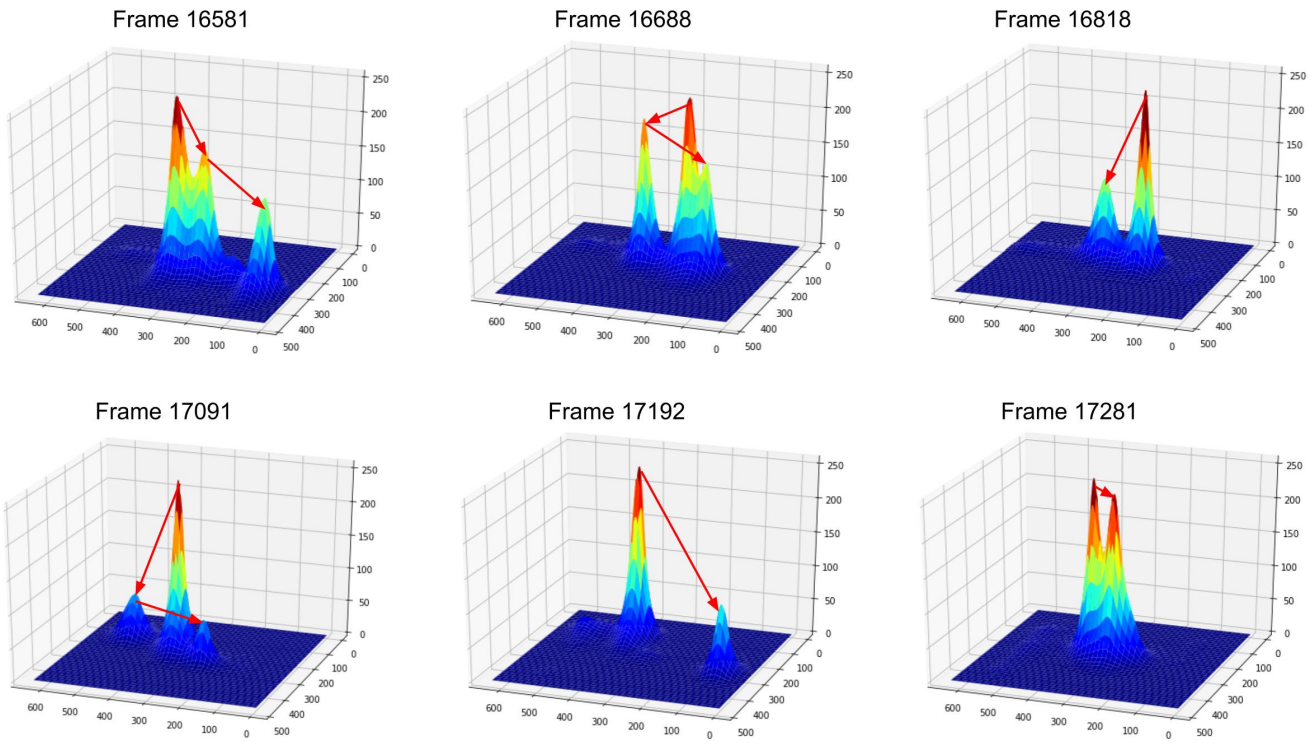

Fig. 20. Extraction of local maxima in Saliency Maps using 0-th dimensional persistent homology for 2D images

probabilistic approach to interpreting the saliency and groundtruth density maps. The KL-Div is formulated as follows:

$$L_3\left(\tilde{\mathbf{y}}, \mathbf{y}^{den}\right) = \sum_i \mathbf{y}_i^{den} \log\left(\frac{\mathbf{y}_i^{den}}{\tilde{\mathbf{y}}_i + \epsilon} + \epsilon\right)$$

The Kullback-Leibler Divergence (KL-Div) measures the difference between the predicted saliency map distribution ($\tilde{\mathbf{y}}$) and the groundtruth density map distribution ($\mathbf{y}^{\text{den}}$) by comparing their information content. The regularization constant (ϵ) is used to prevent the KL-Div from becoming infinity when comparing two unequal distributions. A lower value of KL-Div implies that the predicted saliency map better approximates the groundtruth.

A.3. Using Persistent Homology to predict order of Spatial fixation locations in Saliency Maps

To calculate the priority order of Spatial fixation locations inferred from Saliency Maps, we use a method called Persistent Homology which is used in the field of Topological Data Analysis (TDA) to identify the order in which priority descends within peaks of a 3D projection of Saliency Maps. [36]

Persistent homology provides us with a mathematical framework for capturing and quantifying the significance of topological features, such as peaks of the transposed Saliency Map to 3D, in unprocessed data. This is achieved by considering the super-level sets of a function f, represented as $U_c = f^{-1}([c,1]) = x \in [0,1]^2 : f(x) \geq c$, where $c \in \mathbb{R}$ starts at ∞ and decreases towards $-\infty$. As c decreases, the topology of U_c changes, creating and merging connected components, emerging and filling holes, and so on. Persistent homology tracks the evolution of these topological features in arbitrary dimensions and provides information on their longevity. A feature is considered "more significant" if it lasts longer.

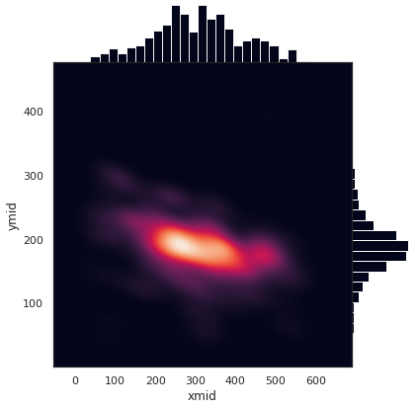

Fig. 21. Kernel Density Estimation plot for Spatial Fixation Locations in a Saliency Map

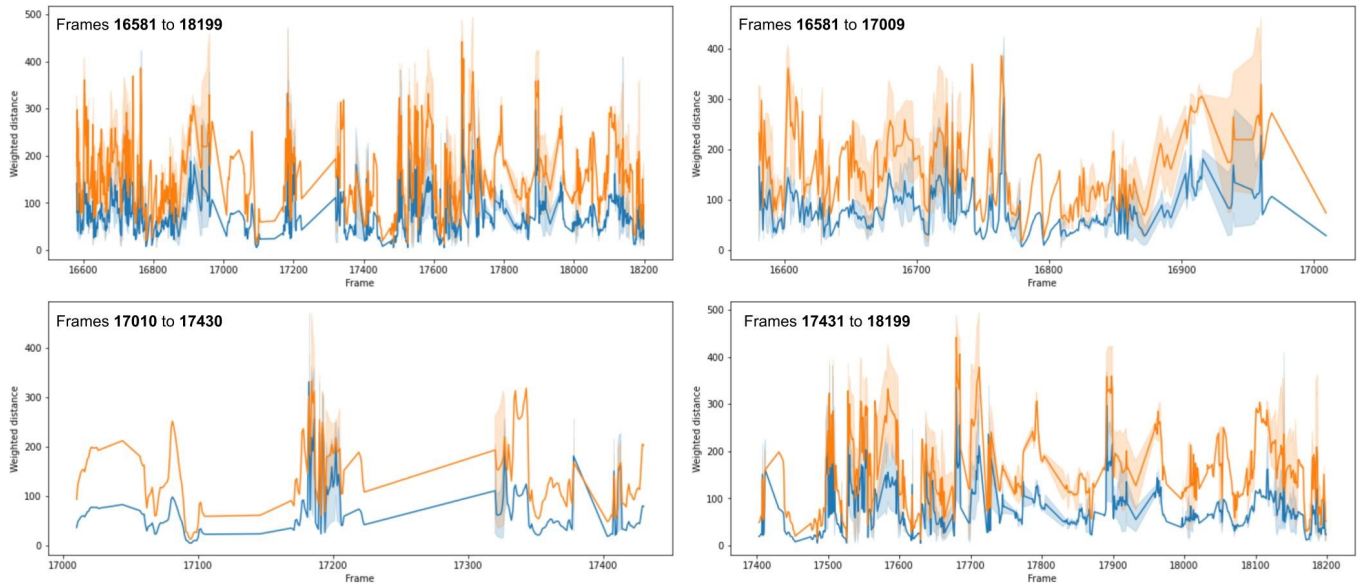

Fig. 22. (Blue) Weighted Euclidean Distance of the Salient Features from Eye Gaze Spatial Coordinates. (Orange) Unweighted Euclidean Distance of all Salient Points from Eye Gaze Spatial Coordinates for Street Crossing Instance

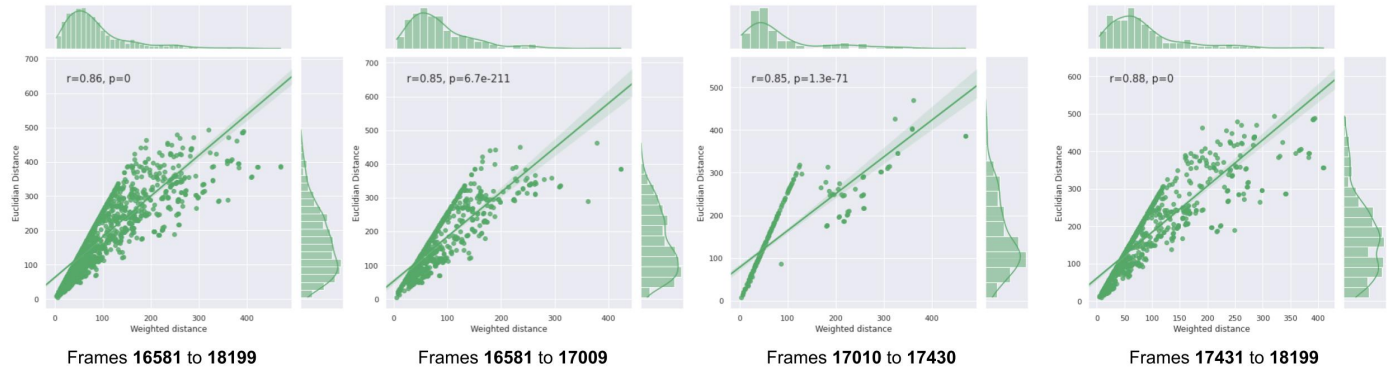

Fig. 23. Correlation of Weighted Distance with Euclidean Distance to understand if the objects gazed in closest proximity by Homonymous Hemaniopic patients are the most Salient or not.

Code for calculating spatial fixation locations using persistent homology

```
def get_persistent_homology(seq):
    peaks = []
    # Maps indices to peaks
    idxtopeak = [None for s in seq]
    # Sequence indices sorted by values
    indices = range(len(seq))
    indices = sorted(indices, key = lambda i:
    seq[i], reverse=True)

    # Process each sample in descending order
    for idx in indices:
        lftdone = (idx > 0 and
        idxtopeak[idx-1] is not None)
        rgtdone = (idx < len(seq)-1 and
        idxtopeak[idx+1] is not None)
        il = idxtopeak[idx-1] if lftdone else None
        ir = idxtopeak[idx+1] if rgtdone else None
        # New peak born
        if not lftdone and not rgtdone:
            peaks.append(Peak(idx))
            idxtopeak[idx] = len(peaks)-1
```

A.4. Analyzing distribution and correlation of Saliency Maps

Through the processed Saliency Maps for Street Crossing 1 instance, we analyze the correlation of the Weighted distance between the Eye Gaze and most salient features on the focal view. It is important to note here that temporal contextual information is important while analyzing temporal movement behavior but since this is a complex task, we haven't taken into account temporal context during analysis. For given N frames within a temporal slice S, the prority of gaze for the most salient features changes once the feature has been gazed. If the temporal slice S of an event is small such that $\lim_{N \to dN} N$, the change in Eye Gaze priority over a frame is the same as the order of Spatial Fixation points in a Saliency Map visualized in 20 for an average of pixels across the frames in the temporal slice S.

Here, we quantify the correlation without considering temporal

saliency of frames. We use the Kernel Density Approach,

$$\hat{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$

using smoothing to analyze the distribution of most Salient Points for Street Crossing Instance. It can be in Fig. 21 observed that the most salient features when concatenated across all time slices, are shifted towards the left of the KDE plot and mainly oriented towards the centroid of the Field of View. Fig. 23 demonstrates the distribution of Weighted Euclidean Distance and Euclidean Distance. We use a modified Mahalanobis Distance to calculate this:

$$d(A, B) = \sqrt{\sum_i \frac{(A_i - B_i)^2}{w_i}},$$

where A_i is the i-th feature for A and w_i is the weight of the pixel i or the inverse variance of the feature i. The utilization of w_i in the denominator is aimed at assigning the lowest Weighted Distance to the point that showcases the greatest saliency and is closest to the Eye Gaze coordinates, and the highest value to the point that is furthest away and displays minimal saliency. Fig. 22 quantifies the correlation between The Weighted Euclidean Distance and the Euclidean Distance and a moderate positive correlation is an indicator that the Eye Gaze proximity was higher at more salient regions than regions with lower saliency.

CONCLUSION

This study bridges computational vision and clinical insight to analyze gaze behavior in individuals with *Homonymous Hemianopia*. By integrating head-mounted eye tracking, IMU-based spatial orientation, and advanced object detection frameworks, we constructed a detailed profile of how HH patients interact with their environments. Our findings reveal the impact of visual field deficits on spatial fixation priorities: HH patients exhibit a marked shift in attention patterns, favoring peripheral over centrally salient regions during complex tasks such as navigating busy streets.

Leveraging algorithms such as Euclidean thresholding and point-in-polygon methods, we quantified gaze-object interactions to uncover consistent behavioral trends. The use of saliency attentive models, incorporating spatial fixation hierarchies and persistent homology, further clarified the deviations in gaze allocation strategies. These results deepen our understanding of HH-related visual adaptations, providing a basis for cost-effective, non-invasive diagnostics and therapeutic advancements.

Future research should explore real-time adaptive gaze-assistive technologies and apply this framework to broader visual impairments, ensuring clinical interventions are as responsive as the challenges they aim to address.

REFERENCES

1. O. Meienberg, W. H. Zangemeister, M. Rosenberg, W. F. Hoyt, and L. Stark, "Saccadic eye movement strategies in patients with homonymous hemianopia," Annals Neurol. **9**, 537–544 (1981).
2. F. Li, S. Munn, and J. Pelz, "A model-based approach to video-based eye tracking," J. Mod. Opt. **55**, 503–531 (2008).
3. S. Fotios, J. Uttley, C. Cheal, and N. Hara, "Using eye-tracking to identify pedestrians' critical visual tasks, part 1. dual task approach," Light. Res. & Technol. **47**, 133–148 (2015).
4. A. Bowers, E. Ananev, A. Mandel, R. Goldstein, and E. Peli, "Driving with hemianopia: Iv. head scanning and detection at intersections in a simulator," Investig. ophthalmology visual science **55** (2014).
5. B. Cesqui, R. Langenberg, van de, F. Lacquaniti, and A. D'Avella, "A novel method for measuring gaze orientation in space in unrestrained head conditions," J. vision **13** (2013).
6. G. Luo, F. Vargas-Martin, and E. Peli, "The role of peripheral vision in saccade planning: Learning from people with tunnel vision," J. vision **8**, 25.1–8 (2008).
7. J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* (2017), pp. 6517–6525.
8. J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," (2018).
9. A. Bochkovskiy, C.-Y. Wang, and H.-y. Liao, "Yolov4: Optimal speed and accuracy of object detection," (2020).
10. R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *2014 IEEE Conference on Computer Vision and Pattern Recognition,* (2014), pp. 580–587.
11. R. Girshick, "Fast r-cnn," in *2015 IEEE International Conference on Computer Vision (ICCV),* (2015), pp. 1440–1448.
12. S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," IEEE Transactions on Pattern Analysis Mach. Intell. **39**, 1137–1149 (2017).
13. A. Ammar, A. Koubaa, M. Ahmed, A. Saad, and B. Benjdira, "Vehicle detection from aerial images using deep learning: A comparative study," Electronics. **10**, 820 (2021).
14. J. W. Kantelhardt, S. A. Zschiegner, E. Koscielny-Bunde, S. Havlin, A. Bunde, and H. Stanley, "Multifractal detrended fluctuation analysis of nonstationary time series," Phys. A: Stat. Mech. its Appl. **316**, 87–114 (2002).
15. X. Li and P. Shang, "Multifractal classification of road traffic flows," Chaos, Solitons Fractals **31**, 1089–1094 (2007).
16. L. R. Miloş, C. Haţiegan, M. C. Miloş, F. M. Barna, and C. Boţoc, "Multifractal detrended fluctuation analysis (mf-dfa) of stock market indexes. empirical evidence from seven central and eastern european markets," Sustainability **12** (2020).
17. D. Makowiec, A. Dudkowska, R. Galaska, A. Rynkiewicz, and J. Wdowczyk, "Monofractality in rr heart rate by multifractal tools," Acta Phys. Polonica B - ACTA PHYS POL B **40** (2009).
18. M. Sharifi, H. Farahani, F. Shahbazi, M. Sharifi, C. Kello, and M. Zare, "Complexity of eye fixation duration time series in reading of persian texts: A multifractal detrended fluctuation analysis," (2017).
19. R. Nigmatullin, S. Dorokhin, and A. Ivchenko, "Generalized hurst hypothesis: Description of time-series in communication systems," Mathematics **9** (2021).
20. R. Rensink, "The dynamic representation of scenes," Vis. Cogn. **7**, 17–42 (2000).
21. V. Setlur, S. Takagi, R. Raskar, M. Gleicher, and B. Gooch, "Automatic image retargeting," in *Proceedings of the 4th International Conference on Mobile and Ubiquitous Multimedia,* (Association for Computing Machinery, New York, NY, USA, 2005), MUM '05, p. 59–68.
22. D. Walther, L. Itti, M. Riesenhuber, T. Poggio, and C. Koch, "Attentional selection for object recognition — a gentle way," in *Biologically Motivated Computer Vision,* H. H. Bülthoff, C. Wallraven, S.-W. Lee, and T. A. Poggio, eds. (Springer Berlin Heidelberg, Berlin, Heidelberg, 2002), pp. 472–479.
23. H. Hadizadeh and I. V. Bajić, "Saliency-aware video compression," IEEE Transactions on Image Process. **23**, 19–33 (2014).
24. V. Mahadevan and N. Vasconcelos, "Biologically inspired object tracking using center-surround saliency mechanisms," IEEE Transactions on Pattern Analysis Mach. Intell. **35**, 541–554 (2013).
25. M. Cornia, L. Baraldi, G. Serra, and R. Cucchiara, "Paying more attention to saliency: Image captioning with saliency and context attention," ACM Trans. Multimed. Comput. Commun. Appl. **14** (2018).
26. M. Cornia, L. Baraldi, G. Serra, and R. Cucchiara, "Predicting human eye fixations via an lstm-based saliency attentive model," IEEE Transactions on Image Process. **27**, 5142–5154 (2018).
27. K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern*

Recognition (CVPR), (2016), pp. 770–778.
28. J. Zhang, Z. Lin, J. Brandt, X. Shen, and S. Sclaroff, "Top-down neural attention by excitation backprop," (2016), pp. 543–559.
29. S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural computation **9**, 1735–80 (1997).
30. J. Donahue, L. A. Hendricks, M. Rohrbach, S. Venugopalan, S. Guadarrama, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," IEEE Transactions on Pattern Analysis Mach. Intell. **39**, 677–691 (2017).
31. A. Karpathy and F. Li, "Deep visual-semantic alignments for generating image descriptions," (2015), pp. 3128–3137.
32. Q. Wu, P. Wang, C. Shen, A. Dick, and A. Van Den Hengel, "Ask me anything: Free-form visual question answering based on knowledge from external sources," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* (2016), pp. 4622–4630.
33. T. Judd, K. Ehinger, F. Durand, and A. Torralba, "Learning to predict where humans look," in *2009 IEEE 12th International Conference on Computer Vision,* (2009), pp. 2106–2113.
34. E. Vig, M. Dorr, and D. Cox, "Large-scale optimization of hierarchical features for saliency prediction in natural images," in *2014 IEEE Conference on Computer Vision and Pattern Recognition,* (2014), pp. 2798–2805.
35. M. Kümmerer, L. Theis, and M. Bethge, "Deep gaze i: Boosting saliency prediction with feature maps trained on imagenet," (2014).
36. S. Huber, "Persistent homology in data science," Data Sci. Anal. Appl. (2021).